

Capstone Project

Certification Project – Finance Me

Banking and Finance Domain

Submitted by: Sumit Kumar Singh

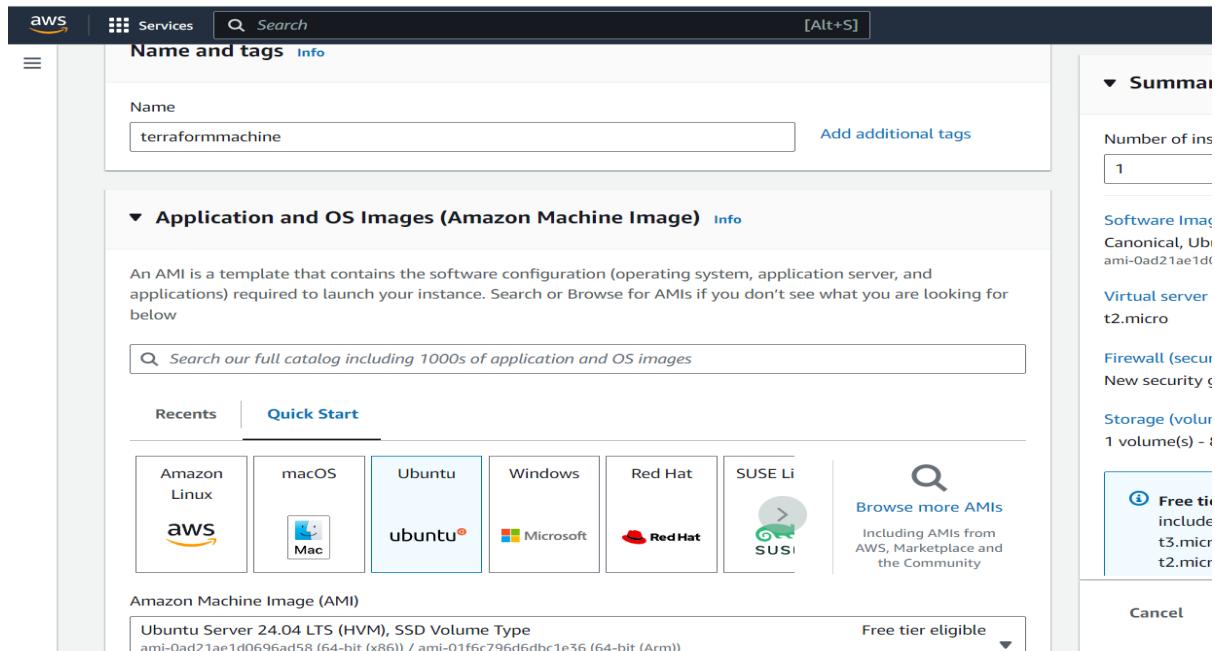
Date Of Submission: 08/08/2024

Submitted to: Mr. Vikul

Banking and finance domain project:

Step1: Go to the AWS console, Launched one instance,in this instance we configure terraform

- Select ubuntu
- In AMI SELECT latest version



- In instance type select t3.medium

Instance type [Info](#) | [Get advice](#)

Instance type

t3.medium

Family: t3 2 vCPU 4 GiB Memory Current generation: true
 On-Demand Windows base pricing: 0.0632 USD per Hour
 On-Demand SUSE base pricing: 0.1011 USD per Hour
 On-Demand RHEL base pricing: 0.0736 USD per Hour
 On-Demand Linux base pricing: 0.0448 USD per Hour

All generations Compare instance types

Additional costs apply for AMIs with pre-installed software

Key pair (login) [Info](#)

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

Key pair name - *required*

project1 [Create new key pair](#)

- In network setting, click on edit, Allow all traffic from anywhere

[Additional charges apply](#) when outside of [free tier allowance](#)

Firewall (security groups) [Info](#)

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

Create security group Select existing security group

We'll create a new security group called '**launch-wizard-19**' with the following rules:

Allow SSH traffic from
Helps you connect to your instance

Anywhere
0.0.0.0/0

Allow HTTPS traffic from the internet
To set up an endpoint, for example when creating a web server

Allow HTTP traffic from the internet
To set up an endpoint, for example when creating a web server

⚠ Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.

- As you can see, an Instance has been created

The screenshot shows the AWS EC2 Instances page. At the top, there's a search bar with 'Find Instance by attribute or tag (case-sensitive)' and a dropdown for 'Instance state'. Below the search bar, there are filters for 'Instance state = running' and 'Clear filters'. The main table lists one instance:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
terraformmachine	i-0b8e3124aec3c7928	Running	t3.medium	Initializing	View alarms +	ap-south-1a

Below the table, the instance details for 'i-0b8e3124aec3c7928 (terraformmachine)' are shown. The 'Details' tab is selected, followed by 'Status and alarms', 'Monitoring', 'Security', 'Networking', 'Storage', and 'Tags'. Under 'Instance summary', the following information is displayed:

- Instance ID: i-0b8e3124aec3c7928 (terraformmachine)
- Public IPv4 address: 52.66.210.235 | [open address](#)
- Private IPv4 addresses: 172.31.39.111
- IPv6 address: -
- Instance state: Running
- Public IPv4 DNS: ec2-52-66-210-235.ap-south-1.compute.amazonaws.com

- Connect the instance

The screenshot shows an EC2 Instance Connect session for the instance 'i-0b8e3124aec3c7928 (terraformmachine)'. The session is connected via SSH. The terminal window displays the following output:

```

aws | Services | Search [Alt+S]
System load: 0.4          Temperature: -273.1 C
Usage of /: 8.3% of 18.33GB Processes: 113
Memory usage: 6%
Swap usage: 0%            Users logged in: 0
                           IPv4 address for ens5: 172.31.39.111

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/*copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-39-111:~$ 

```

At the bottom of the terminal window, it says:

i-0b8e3124aec3c7928 (terraformmachine)
Public IPs: 52.66.210.235 Private IPs: 172.31.39.111

Step 2: Execute below commands in instances

\$sudo su

```
ubuntu@ip-172-31-39-111:~$ sudo su
root@ip-172-31-39-111:/home/ubuntu# □

i-0b8e3124aec3c7928 (terraformmachine)
Public IPs: 52.66.210.235 Private IPs: 172.31.39.111
```

Step 3: Now Search [terraform.io](#) in browser

- Click on download



- Scroll down come to linux package manager, there is some code need to execute one by one for install terraform

Linux

Package manager

[Ubuntu/Debian](#) [CentOS/RHEL](#) [Fedora](#) [Amazon Linux](#) [Homebrew](#)

```
wget -O- https://apt.releases.hashicorp.com/gpg | sudo gpg --dearmor -o /usr/share/keyrings/hashicorp-archive-keyring.gpg
echo "deb [signed-by=/usr/share/keyrings/hashicorp-archive-keyring.gpg] https://apt.releases.hashicorp.com/ $(lsb_release -cs) main" | sudo tee /etc/apt/sources.list.d/hashicorp.list
sudo apt update && sudo apt install terraform
```

Binary download

200

ARM64

- Paste first below command

```
$wget -O- https://apt.releases.hashicorp.com/gpg | sudo gpg -  
-dearmor -o /usr/share/keyrings/hashicorp-archive-  
keyring.gpg
```

```
ubuntu@ip-172-31-39-111:~$ sudo su  
root@ip-172-31-39-111:/home/ubuntu# wget -O- https://apt.releases.hashicorp.com/gpg | sudo gpg --dearmor -o /usr/share/keyrings/hashicorp-archive-keyring.gpg  
--2024-08-06 06:58:25-- https://apt.releases.hashicorp.com/gpg  
Resolving apt.releases.hashicorp.com (apt.releases.hashicorp.com)... 18.172.78.129, 18.172.78.30, 18.172.78.65, ...  
Connecting to apt.releases.hashicorp.com (apt.releases.hashicorp.com)|18.172.78.129|:443... connected.  
HTTP request sent, awaiting response... 200 OK  
Length: 3980 (3.9K) [binary/octet-stream]  
Saving to: 'STDOUT'  
  
- 100%[=====] 3.89K ---KB/s in 0s  
  
2024-08-06 06:58:25 (862 MB/s) - written to stdout [3980/3980]  
root@ip-172-31-39-111:/home/ubuntu# []  
  
i-0b8e3124aec3c7928 (terraformmachine)  
PublicIPs: 52.66.210.235 PrivateIPs: 172.31.39.111
```

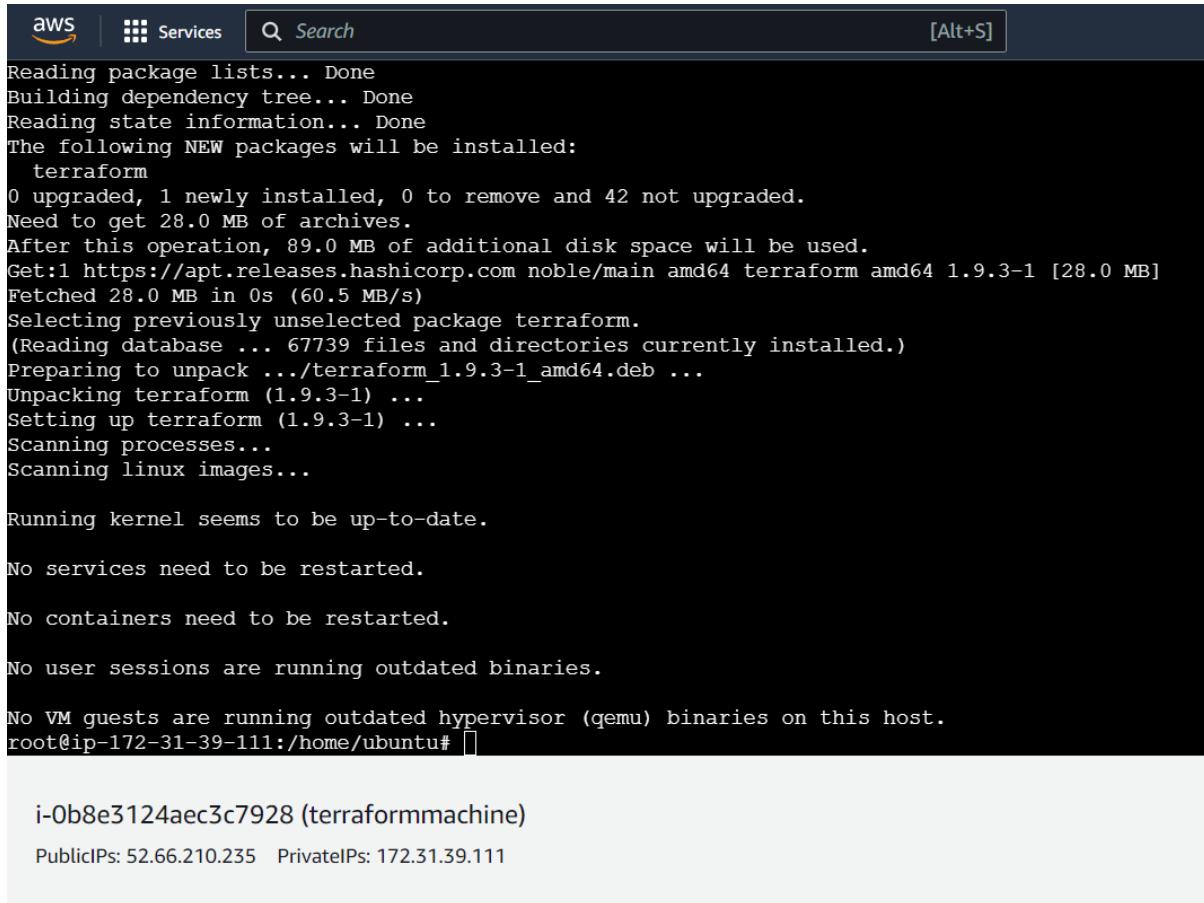
- Paste Second below command

```
$echo "deb [signed-by=/usr/share/keyrings/hashicorp-  
archive-keyring.gpg] https://apt.releases.hashicorp.com  
$(lsb_release -cs) main" | sudo tee  
/etc/apt/sources.list.d/hashicorp.list
```

```
aws | Services | Q Search [Alt+S]  
  
The programs included with the Ubuntu system are free software;  
the exact distribution terms for each program are described in the  
individual files in /usr/share/doc/*copyright.  
  
Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by  
applicable law.  
  
To run a command as administrator (user "root"), use "sudo <command>".  
See "man sudo_root" for details.  
  
ubuntu@ip-172-31-39-111:~$ sudo su  
root@ip-172-31-39-111:/home/ubuntu# wget -O- https://apt.releases.hashicorp.com/gpg | sudo gpg --dearmor -o /usr/share  
--2024-08-06 06:58:25-- https://apt.releases.hashicorp.com/gpg  
Resolving apt.releases.hashicorp.com (apt.releases.hashicorp.com)... 18.172.78.129, 18.172.78.30, 18.172.78.65, ...  
Connecting to apt.releases.hashicorp.com (apt.releases.hashicorp.com)|18.172.78.129|:443... connected.  
HTTP request sent, awaiting response... 200 OK  
Length: 3980 (3.9K) [binary/octet-stream]  
Saving to: 'STDOUT'  
  
- 100%[=====]  
  
2024-08-06 06:58:25 (862 MB/s) - written to stdout [3980/3980]  
  
root@ip-172-31-39-111:/home/ubuntu# echo "deb [signed-by=/usr/share/keyrings/hashicorp-archive-keyring.gpg] https://ap  
main" | sudo tee /etc/apt/sources.list.d/hashicorp.list  
deb [signed-by=/usr/share/keyrings/hashicorp-archive-keyring.gpg] https://apt.releases.hashicorp.com noble main  
root@ip-172-31-39-111:/home/ubuntu# []  
  
i-0b8e3124aec3c7928 (terraformmachine)  
PublicIPs: 52.66.210.235 PrivateIPs: 172.31.39.111
```

- Now Paste Third below command

\$sudo apt update && sudo apt install terraform



```

aws | Services | Search [Alt+S]
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following NEW packages will be installed:
  terraform
0 upgraded, 1 newly installed, 0 to remove and 42 not upgraded.
Need to get 28.0 MB of archives.
After this operation, 89.0 MB of additional disk space will be used.
Get:1 https://apt.releases.hashicorp.com noble/main amd64 terraform amd64 1.9.3-1 [28.0 MB]
Fetched 28.0 MB in 0s (60.5 MB/s)
Selecting previously unselected package terraform.
(Reading database ... 67739 files and directories currently installed.)
Preparing to unpack .../terraform_1.9.3-1_amd64.deb ...
Unpacking terraform (1.9.3-1) ...
Setting up terraform (1.9.3-1) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

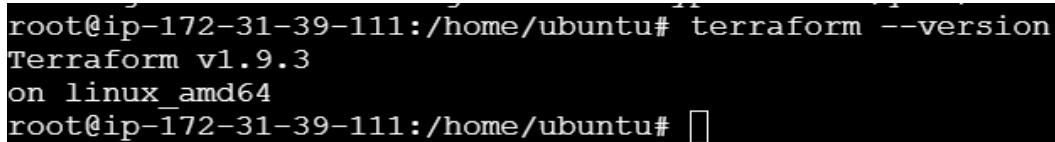
No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@ip-172-31-39-111:/home/ubuntu# 
```

i-0b8e3124aec3c7928 (terraformmachine)
 PublicIPs: 52.66.210.235 PrivateIPs: 172.31.39.111

Step 4: Now execute below command For check terraform version

\$terraform - --version



```

root@ip-172-31-39-111:/home/ubuntu# terraform --version
Terraform v1.9.3
on linux_amd64
root@ip-172-31-39-111:/home/ubuntu# 
```

i-0b8e3124aec3c7928 (terraformmachine)
 PublicIPs: 52.66.210.235 PrivateIPs: 172.31.39.111

- You can see we have installed terraform

- Step 5: Now we will Create a directory by execute below command

\$mkdir project

\$cd project

```
root@ip-172-31-39-111:/home/ubuntu# mkdir project
root@ip-172-31-39-111:/home/ubuntu# ls
project
root@ip-172-31-39-111:/home/ubuntu# cd project/
root@ip-172-31-39-111:/home/ubuntu/project#
```

i-0b8e3124aec3c7928 (terraformmachine)

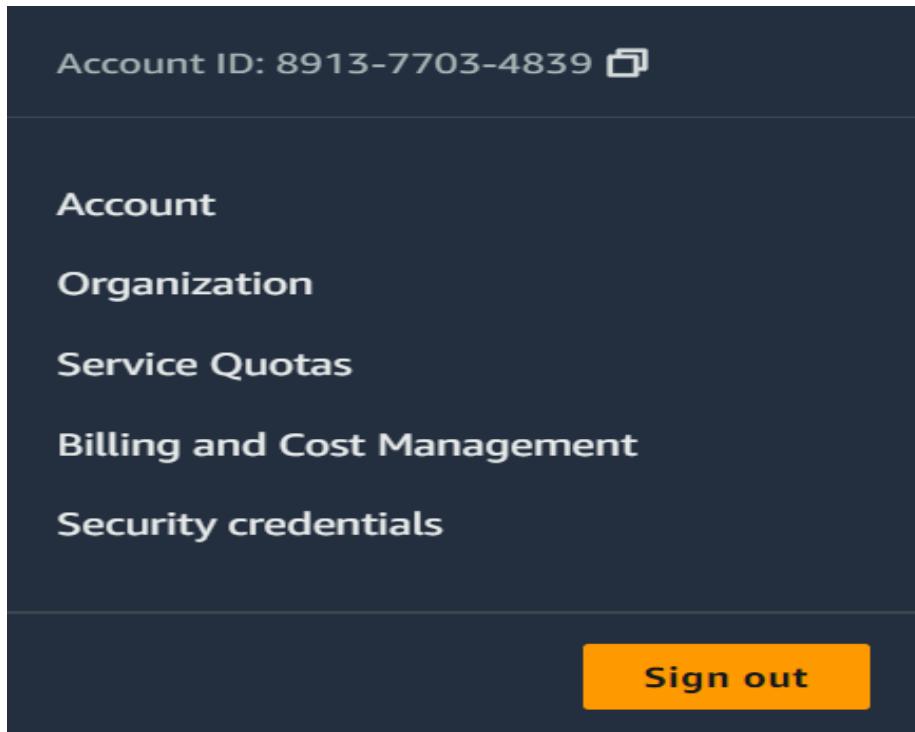
Public IPs: 52.66.210.235 Private IPs: 172.31.39.111

Step 6: Now we will generate the access key and secret access key

- Go to aws console page,click on upper right corner



- Now click on security credentials



- Scroll down there is a option of Access keys,click on create access key

Access keys (1)					Actions	Create access key
Access key ID	Created on	Access key last used	Region last used	Service last us		
AKIA47CRWEZLVO7QFNMU	7 days ago	7 days ago	us-east-1	ec2	<input checked="" type="radio"/>	

- Now tick on check box and again click on create access key

Alternatives to root user access keys Info

⚠ Root user access keys are not recommended
We don't recommend that you create root user access keys. Because you can't specify the root user in a permissions policy, you can't limit its permissions, which is a best practice.
Instead, use alternatives such as an IAM role or a user in IAM Identity Center, which provide temporary rather than long-term credentials. [Learn More](#)

If your use case requires an access key, create an IAM user with an access key and apply least privilege permissions for that user. [Learn More](#)

Continue to create access key?

I understand creating a root access key is not a best practice, but I still want to create one.

[Cancel](#) [Create access key](#)

- You can see I have successfully generated the access key

The screenshot shows the 'Access key' section of the AWS IAM console. It displays two access keys: 'Access key' (AKIA47CRWEZLZML2VY4G) and 'Secret access key' (a masked string). A 'Show' link is available for the secret key.

Step 7: Now again go to terraform.io page on the browser

- Click on download registry

The screenshot shows the Terraform Community homepage. The navigation bar includes links for Overview, Use Cases, Registry, Tutorials, Docs, and Community. The main content area is currently empty.

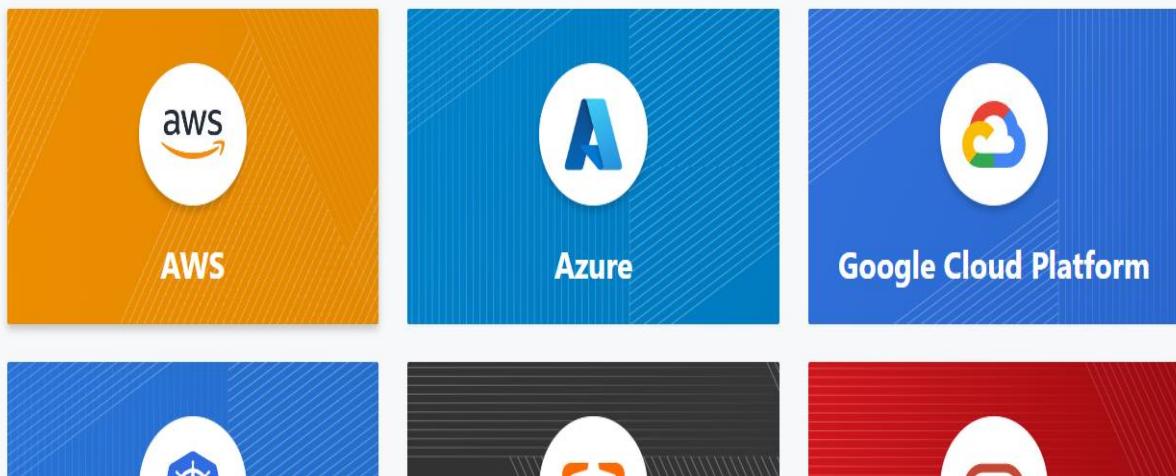
- Click on browse provider

The screenshot shows the Terraform Registry homepage. A red box highlights the 'Browse Providers' button. Other buttons include 'Browse Modules' and 'Browse'. The text '4343 providers, 17067 modules & 82' is visible at the bottom.

- Click on AWS

Providers

Providers are a logical abstraction of an upstream API. They are responsible for understanding API interactions and exposing resources.



- **Click on Documentation**

The screenshot shows the HashiCorp Registry provider documentation for the AWS provider. The URL is [Providers / hashicorp / aws / Version 5.59.0](#). The page has a sidebar on the left with "AWS DOCUMENTATION" and a search bar. The main content area is titled "AWS Provider" and contains a brief description: "Use the Amazon Web Services (AWS) provider to interact with the many resources supported by AWS. You must configure the provider with the proper credentials before you can use it." It also features a "Multi-language provider docs" section, a "Terraform" dropdown, and a "ON THIS PAGE" link.

- **Scroll down and go to the usage section ,copy the code for configure access key and secret access key)**

Usage:

```
provider "aws" {  
    region      = "us-west-2"  
    access_key = "my-access-key"  
    secret_key = "my-secret-key"  
}
```

[Copy](#)

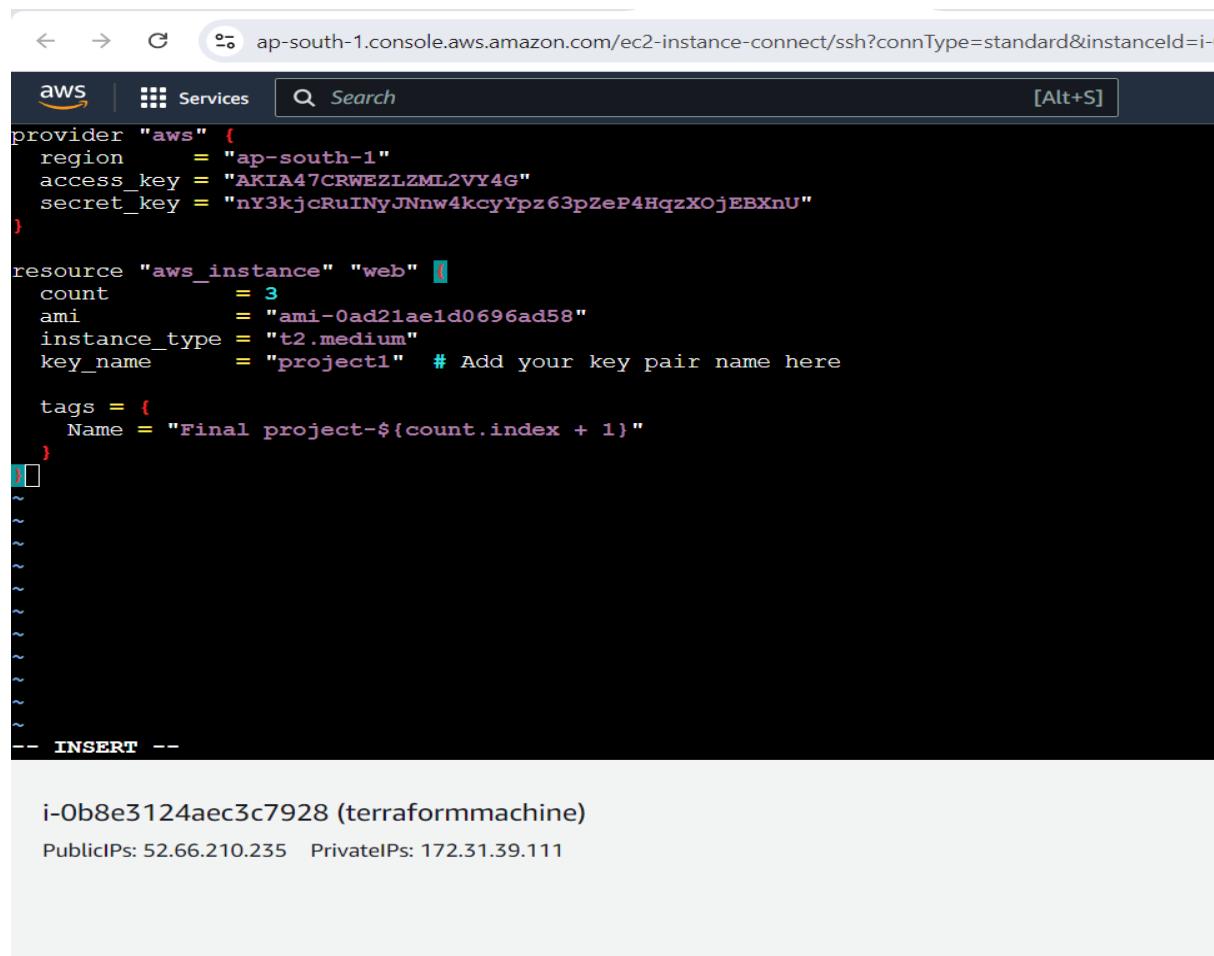
Other settings related to authorization can be configured, such as:

- [profile](#)

Step 8: Create a file with name ec2.tf in instance by execute below command(terraform file is always ends with .tf)

\$vi ec2.tf

- In this we use provider as aws
- I have pasted my access key and secret access key
- In resource we write aws instance
- Count as 3(because we are creating three instances)
- Pasted ubuntu ami id
- In instance type we write t3.medium
- And in tag we write the instance name



The screenshot shows a terminal window within an AWS EC2 Instance Connect session. The URL in the browser bar is ap-south-1.console.aws.amazon.com/ec2-instance-connect/ssh?connType=standard&instanceId=i-0b8e3124aec3c7928. The terminal window has a dark background and displays the following Terraform configuration:

```
provider "aws" {
  region      = "ap-south-1"
  access_key  = "AKIA47CRWEZLZML2VY4G"
  secret_key  = "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"
}

resource "aws_instance" "web" {
  count       = 3
  ami         = "ami-0ad21ae1d0696ad58"
  instance_type = "t2.medium"
  key_name    = "project1" # Add your key pair name here

  tags = {
    Name = "Final project-${count.index + 1}"
  }
}

-- INSERT --
```

Below the code, the terminal shows the instance ID and its network details:

```
i-0b8e3124aec3c7928 (terraformmachine)
PublicIPs: 52.66.210.235 PrivateIPs: 172.31.39.111
```

- Press **esc--->:wq**

Step 9: Now execute below command for initialize the terraform

\$terraform init

The screenshot shows a terminal window within the AWS CloudShell interface. The terminal title is 'ap-south-1.console.aws.amazon.com/ec2-instance-connect/ssh?connType=standard&instanceId=i-0b8e3124aec3c7928'. The terminal header includes the AWS logo, 'Services' button, search bar, and '[Alt+S]' key binding. The terminal content displays the output of the 'terraform init' command:

```
Terraform v1.9.3
on linux_amd64
root@ip-172-31-39-111:/home/ubuntu# mkdir project
root@ip-172-31-39-111:/home/ubuntu# ls
project
root@ip-172-31-39-111:/home/ubuntu# cd project/
root@ip-172-31-39-111:/home/ubuntu/project# vi ec2.tf
root@ip-172-31-39-111:/home/ubuntu/project# terraform init
Initializing the backend...
Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v5.61.0...
- Installed hashicorp/aws v5.61.0 (signed by HashiCorp)
Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

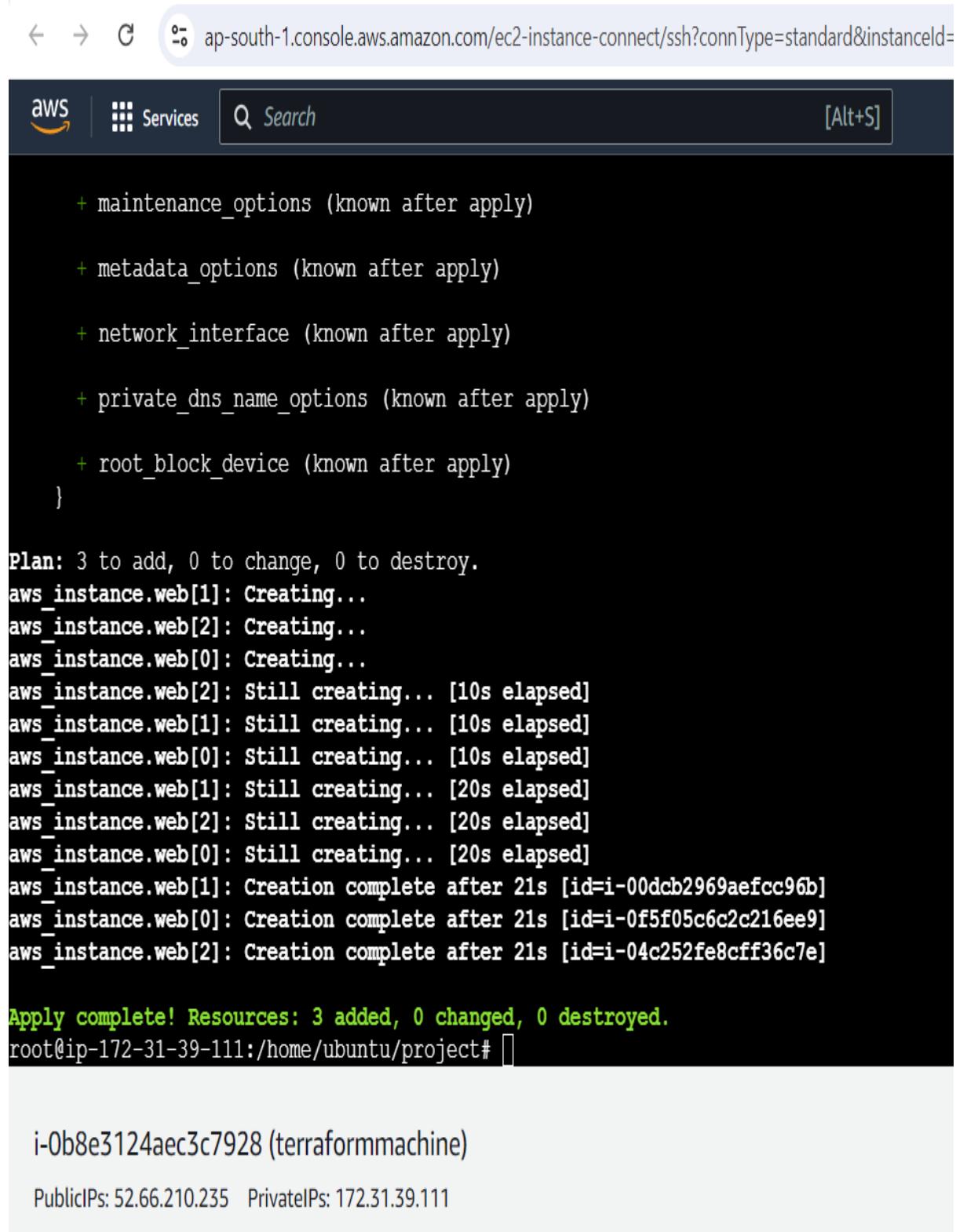
You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
root@ip-172-31-39-111:/home/ubuntu/project#
```

At the bottom of the terminal, the instance ID 'i-0b8e3124aec3c7928 (terraformmachine)' and its network details 'PublicIPs: 52.66.210.235 PrivateIPs: 172.31.39.111' are displayed.

Step 10: Now execute below command to execute ec2.tf file

\$terraform apply - -auto-approve



```
← → C ap-south-1.console.aws.amazon.com/ec2-instance-connect/ssh?connType=standard&instanceId= [Alt+S]

aws Services Search [Alt+S]

+ maintenance_options (known after apply)
+ metadata_options (known after apply)
+ network_interface (known after apply)
+ private_dns_name_options (known after apply)
+ root_block_device (known after apply)
}

Plan: 3 to add, 0 to change, 0 to destroy.
aws_instance.web[1]: Creating...
aws_instance.web[2]: Creating...
aws_instance.web[0]: Creating...
aws_instance.web[2]: Still creating... [10s elapsed]
aws_instance.web[1]: Still creating... [10s elapsed]
aws_instance.web[0]: Still creating... [10s elapsed]
aws_instance.web[1]: Still creating... [20s elapsed]
aws_instance.web[2]: Still creating... [20s elapsed]
aws_instance.web[0]: Still creating... [20s elapsed]
aws_instance.web[1]: Creation complete after 21s [id=i-00dcb2969aefcc96b]
aws_instance.web[0]: Creation complete after 21s [id=i-0f5f05c6c2c216ee9]
aws_instance.web[2]: Creation complete after 21s [id=i-04c252fe8cff36c7e]

Apply complete! Resources: 3 added, 0 changed, 0 destroyed.
root@ip-172-31-39-111:/home/ubuntu/project# []

i-0b8e3124aec3c7928 (terraformmachine)

PublicIPs: 52.66.210.235 PrivateIPs: 172.31.39.111
```

- You can see we have added 3 resources

Step 11: For verify instance has been created go to ec2 dashboard, there you can see we have created 3 instances via terraform

- Rename the instance as Final-project-Master(In which we install Jenkins)
- Final-project-Host
- Final-project-Monitoring(Prometheus and Grafana we install in this)

Instances (1/9) Info							
		Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
<input type="checkbox"/> Name ↴		Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
<input type="checkbox"/>	host-new-with key	i-0bf6c70cf5cf0ec99	Stopped	t2.micro	-	View alarms +	ap-south-1b
<input checked="" type="checkbox"/>	Final project-1	i-0f5f05c6c2c216ee9	Running	t2.medium	2/2 checks passed	View alarms +	ap-south-1b
<input type="checkbox"/>	Final project-2	i-00dc2969aefcc96b	Running	t2.medium	2/2 checks passed	View alarms +	ap-south-1b
<input type="checkbox"/>	Final project-3	i-04c252fe8cff36c7e	Running	t2.medium	2/2 checks passed	View alarms +	ap-south-1b
<input type="checkbox"/>	terraform	i-0a2c6eef1b959eab3	Stopped	t2.micro	-	View alarms +	ap-south-1a
<input type="checkbox"/>	terrafrommachine	i-0b8e3124aec3c7928	Running	t3.medium	2/2 checks passed	View alarms +	ap-south-1a

- Rename created instance

Instances (1/4) Info							
		Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
<input type="checkbox"/> Name ↴		Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
<input type="checkbox"/>	Final-project-Mater	i-0f5f05c6c2c216ee9	Running	t2.medium	2/2 checks passed	View alarms +	ap-south-1b
<input type="checkbox"/>	Final-project-Monitoring	i-00dc2969aefcc96b	Running	t2.medium	2/2 checks passed	View alarms +	ap-south-1b
<input checked="" type="checkbox"/>	Final-project-Host	i-04c252fe8cff36c7e	Running	t2.medium	2/2 checks passed	View alarms +	ap-south-1b
<input type="checkbox"/>	terrafrommachine	i-0b8e3124aec3c7928	Running	t3.medium	2/2 checks passed	View alarms +	ap-south-1a

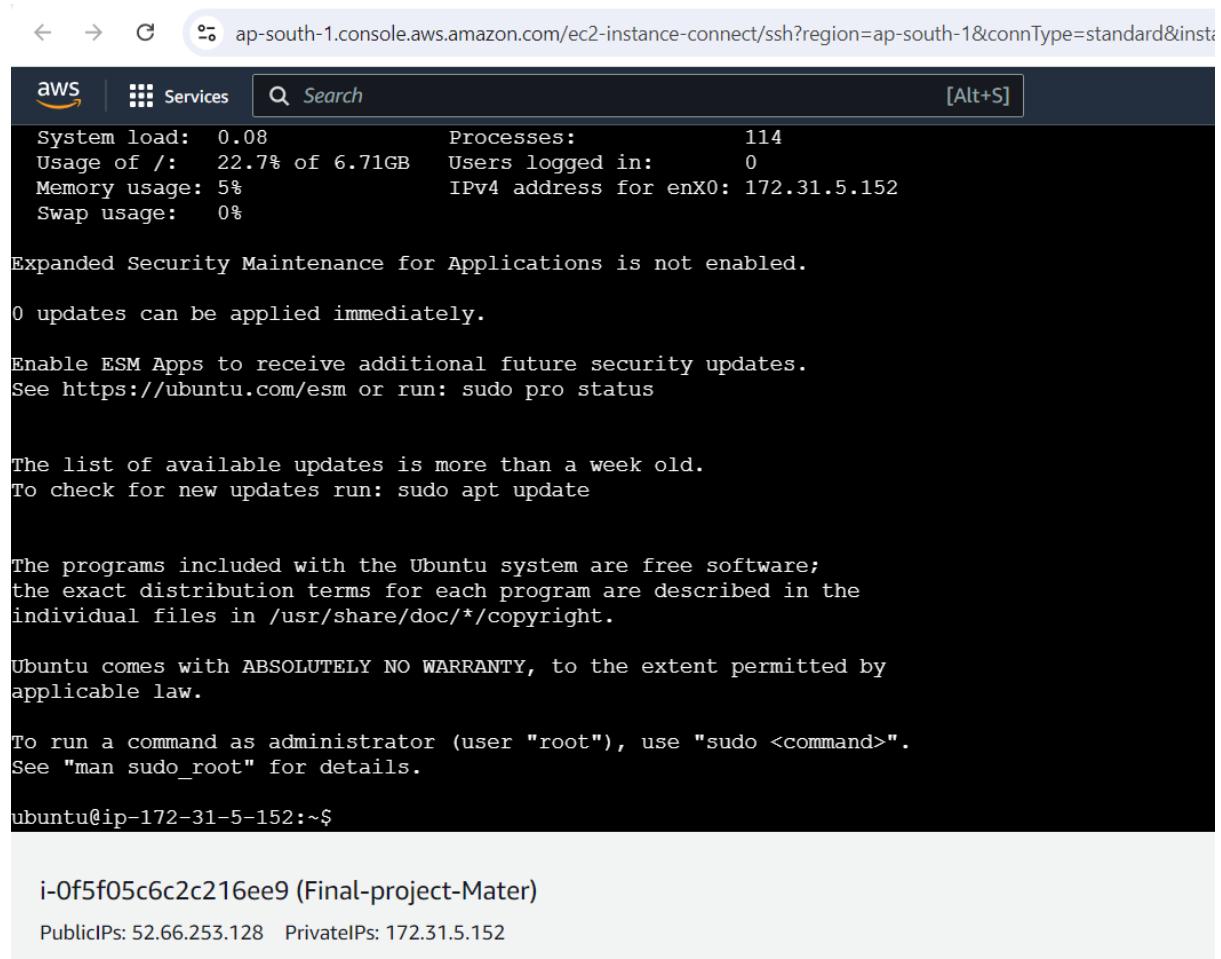
Step 12: Now edited the security group of all three instances as below

- Go to inbound rules delete existing rule and add new one

Inbound rules	Security group rule ID	Type	Protocol	Port range	Source	Description - optional
	-	All traffic	All	All	Anywhere...	For project

Cancel [Preview changes](#) [Save rules](#)

Step 13: Now connect the final-project-master instance



The screenshot shows a terminal window titled "aws Services" with a search bar and a keyboard shortcut "[Alt+S]". The terminal displays various system statistics:

- System load: 0.08
- Processes: 114
- Usage of /: 22.7% of 6.71GB
- Users logged in: 0
- Memory usage: 5%
- IPv4 address for enX0: 172.31.5.152
- Swap usage: 0%

Below the stats, it says "Expanded Security Maintenance for Applications is not enabled." and "0 updates can be applied immediately." It also provides instructions to "Enable ESM Apps to receive additional future security updates." and "See <https://ubuntu.com/esm> or run: sudo pro status".

The terminal then notes that "The list of available updates is more than a week old." and "To check for new updates run: sudo apt update".

It includes a copyright notice: "The programs included with the Ubuntu system are free software; the exact distribution terms for each program are described in the individual files in /usr/share/doc/*copyright".

It states that "Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by applicable law."

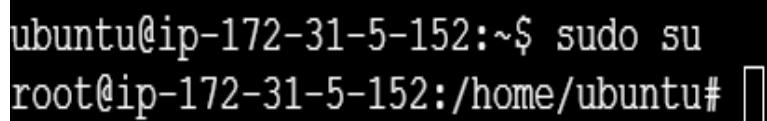
To run commands as root, it suggests using "sudo <command>" and "See "man sudo_root" for details."

The prompt "ubuntu@ip-172-31-5-152:~\$" is shown at the end of the terminal session.

At the bottom of the terminal window, the instance ID "i-0f5f05c6c2c216ee9 (Final-project-Mater)" and its PublicIPs and PrivateIPs are listed.

- Now execute below command for become root user

\$sudo su



```
ubuntu@ip-172-31-5-152:~$ sudo su
root@ip-172-31-5-152:/home/ubuntu# 
```

i-0f5f05c6c2c216ee9 (Final-project-Mater)

PublicIPs: 52.66.253.128 PrivateIPs: 172.31.5.152

Step 14: Now we will install Jenkins in this machine ,for install Jenkins followed below steps

- Go to deployment script github repository
- Click on Jenkins.sh file(it is a shell script file to install Jenkins)



The screenshot shows a GitHub repository interface. On the left, there's a sidebar with a 'Go to file' search bar and a list of files: jenkins.sh (selected), k8s-master.sh, k8s-nodes.sh, and readme-k8s. The main area displays the contents of the jenkins.sh file. The file is a shell script with 17 lines of code. The code installs Java, adds the Jenkins keyring, and configures the Jenkins service.

```
#!/bin/bash
# USE UBUNTU20.04 - INSTANCE: 2GB RAM + 2VCPU MIN - WILL ONLY WORK
sudo apt update -y
sudo apt install openjdk-11-jdk -y
sudo apt update -y
sudo apt install maven -y
curl -fsSL https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key | sudo tee \
  /usr/share/keyrings/jenkins-keyring.asc > /dev/null
echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] \
  https://pkg.jenkins.io/debian-stable binary/ | sudo tee \
  /etc/apt/sources.list.d/jenkins.list > /dev/null
sudo apt update -y
sudo apt install jenkins -y
service jenkins start
cat /var/lib/jenkins/secrets/initialAdminPassword
#chmod 777 jenkins.sh
./jenkins.sh
```

- Click on the raw button and copy the link

← → ⌂ raw.githubusercontent.com/sumitsingh231/Deployment-script/main/jenkins.sh

```
#!/bin/bash
# USE UBUNTU20.04 - INSTANCE: 2GB RAM + 2VCPU MIN - WILL ONLY WORK
sudo apt update -y
sudo apt install openjdk-11-jdk -y
sudo apt update -y
sudo apt install maven -y
curl -fsSL https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key | sudo tee \
  /usr/share/keyrings/jenkins-keyring.asc > /dev/null
echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] \
  https://pkg.jenkins.io/debian-stable binary/ | sudo tee \
  /etc/apt/sources.list.d/jenkins.list > /dev/null
sudo apt update -y
sudo apt install jenkins -y
service jenkins start
cat /var/lib/jenkins/secrets/initialAdminPassword
#chmod 777 jenkins.sh
./jenkins.sh
```

Step 15: Now execute below command in master machine

\$wget <paste the copied link>

\$wget

<https://raw.githubusercontent.com/sumitsingh231/Deployment-script/main/jenkins.sh>

```
ubuntu@ip-172-31-5-152:~$ sudo su
root@ip-172-31-5-152:/home/ubuntu# wget https://raw.githubusercontent.com/sumitsingh231/Deployment-script/main/jenkins.sh
--2024-08-06 07:31:40-- https://raw.githubusercontent.com/sumitsingh231/Deployment-script/main/jenkins.sh
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.111.133, 185.199.108.133, 185.199.109.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.111.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 642 [text/plain]
Saving to: 'jenkins.sh'

jenkins.sh          100%[=====] 642 --.-KB/s   in 0s

2024-08-06 07:31:40 (27.6 MB/s) - 'jenkins.sh' saved [642/642]
root@ip-172-31-5-152:/home/ubuntu# 
```

i-Of5f05c6c2c216ee9 (Final-project-Mater)
PublicIPs: 52.66.253.128 PrivateIPs: 172.31.5.152

Step 16: Now change the mode for giving read write execute permission for this execute below command

\$chmod +x jenkins.sh

```
root@ip-172-31-5-152:/home/ubuntu# chmod +x jenkins.sh
root@ip-172-31-5-152:/home/ubuntu# ls
jenkins.sh
root@ip-172-31-5-152:/home/ubuntu# ls -l
total 4
-rwxr-xr-x 1 root root 642 Aug  6 07:31 jenkins.sh
root@ip-172-31-5-152:/home/ubuntu# 
```

i-Of5f05c6c2c216ee9 (Final-project-Mater)

PublicIPs: 52.66.253.128 PrivateIPs: 172.31.5.152

Step 17: Now execute below command to run the Jenkins.sh shell script

\$./jenkins.sh

The screenshot shows a terminal window within the AWS CloudShell interface. The URL at the top is ap-south-1.console.aws.amazon.com/ec2-instance-connect/ssh?region=ap-south-1&connType=standard&instanceId=i-0f5f05c6c2c216ee9&osUser=ubuntu. The terminal header includes the AWS logo, services menu, search bar, and Alt+S keybinding.

```
Fetched 92.1 MB in 6s (14.3 MB/s)
Selecting previously unselected package net-tools.
(Reading database ... 71368 files and directories currently installed.)
Preparing to unpack .../net-tools_2.10-0.1ubuntu4_amd64.deb ...
Unpacking net-tools (2.10-0.1ubuntu4) ...
Selecting previously unselected package jenkins.
Preparing to unpack .../jenkins_2.452.3_all.deb ...
Unpacking jenkins (2.452.3) ...
Setting up net-tools (2.10-0.1ubuntu4) ...
Setting up jenkins (2.452.3) ...
Created symlink /etc/systemd/system/multi-user.target.wants/jenkins.service → /usr/lib/systemd/system/jenkins.service.
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

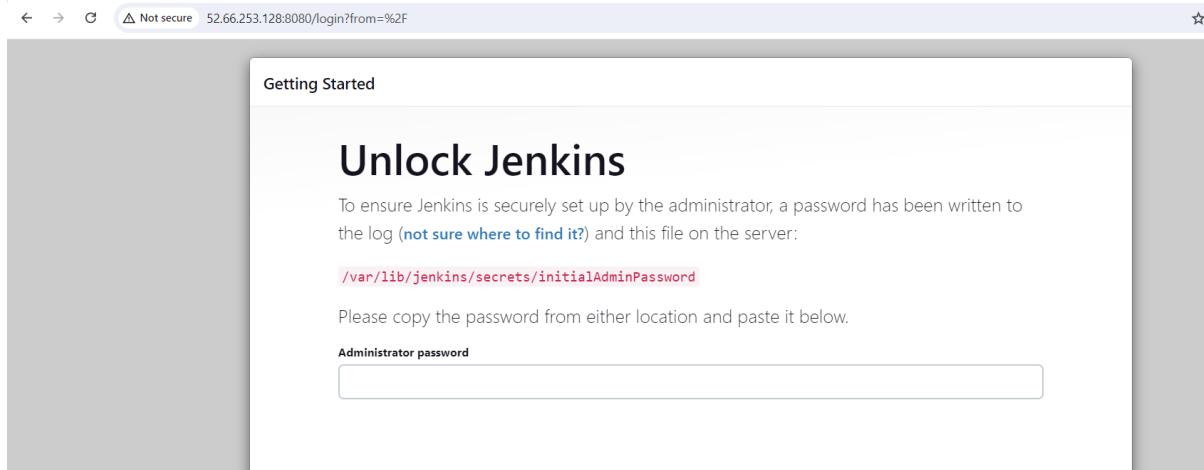
No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
6b8b68f684a043abbf33c504137cad79
root@ip-172-31-5-152:/home/ubuntu# jenkins --version
2.452.3
root@ip-172-31-5-152:/home/ubuntu# []
```

i-0f5f05c6c2c216ee9 (Final-project-Mater)
PublicIPs: 52.66.253.128 PrivateIPs: 172.31.5.152

- You can see we have installed Jenkins

Step 18: Now copy the public ip of Jenkins master machine and paste it into the browser with:8080(Jenkins default port)



Step 19: copy the highlighted path and execute below command to see administrator password

\$cat /var/lib/jenkins/secrets/initialAdminPassword

```
No VM guests are running outdated hypervisor (qemu) binaries on this host.  
4dba5b4af9904cfcac2bb80ccae840f4  
root@ip-172-31-10-193:/home/ubuntu# jenkins --version  
2.452.3  
root@ip-172-31-10-193:/home/ubuntu# cat /var/lib/jenkins/secrets/initialAdminPassword  
4dba5b4af9904cfcac2bb80ccae840f4  
root@ip-172-31-10-193:/home/ubuntu#
```

i-05a5c378abcf815a3 (Final project-master)

Public IPs: 13.233.144.39 Private IPs: 172.31.10.193

- Paste the password into administrator password section then click on continue**

Unlock Jenkins

To ensure Jenkins is securely set up by the administrator, a password has been written to the log ([not sure where to find it?](#)) and this file on the server:

```
/var/lib/jenkins/secrets/initialAdminPassword
```

Please copy the password from either location and paste it below.

Administrator password

```
.....
```

Continue

- Click on install suggested plugins

Customize Jenkins

Plugins extend Jenkins with additional features to support many different needs.

Install suggested plugins

Install plugins the Jenkins community finds most useful.

Select plugins to install

Select and install plugins most suitable for your needs.

- You can see plugins has been installed and Jenkins getting started,in this page used user name and password as admin

Getting Started

Username	<input type="text" value="admin"/>
Password	<input type="password" value="....."/>
Confirm password	<input type="password" value="....."/>
Full name	<input type="text" value="admin"/>
E-mail address	<input type="text" value="sssumitaws@gmail.com"/>

Jenkins 2.452.3 [Skip and continue as admin](#) [Save and Continue](#)

- Now click on save and finish

Getting Started

Instance Configuration

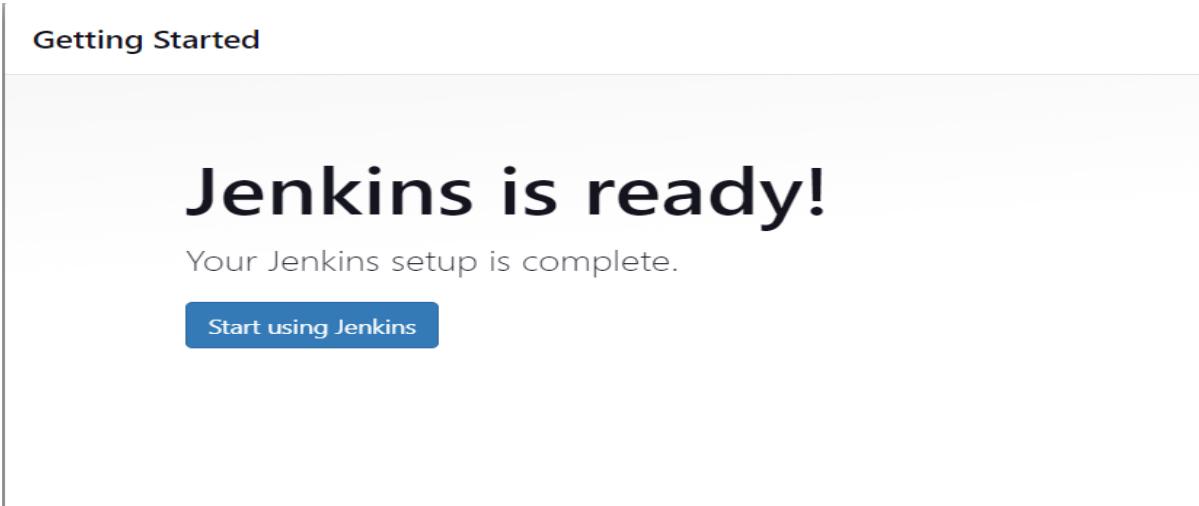
Jenkins URL:	<input type="text" value="http://52.66.253.128:8080/"/>
--------------	---

The Jenkins URL is used to provide the root URL for absolute links to various Jenkins resources. That means this value is required for proper operation of many Jenkins features including email notifications, PR status updates, and the `BUILD_URL` environment variable provided to build steps.

The proposed default value shown is **not saved yet** and is generated from the current request, if possible. The best practice is to set this value to the URL that users are expected to use. This will avoid confusion when sharing or viewing links.

Jenkins 2.452.3 [Not now](#) [Save and Finish](#)

- You can see now Jenkins is ready, now click on start using jenkins

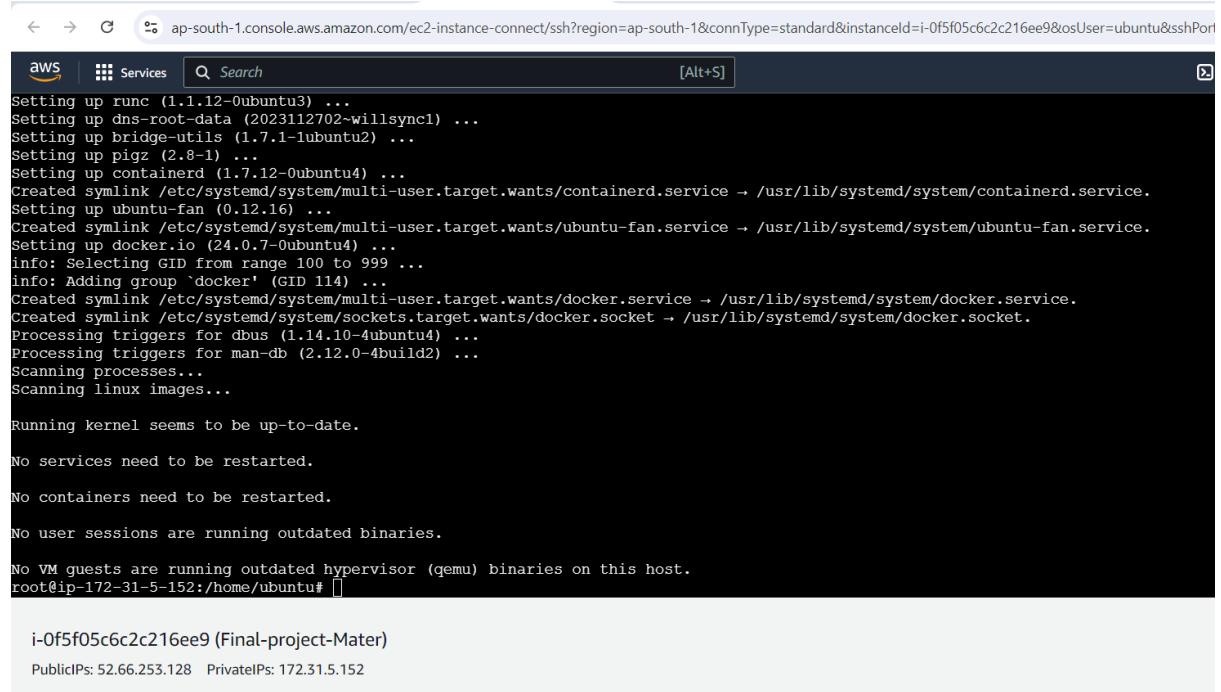


- Now you can see the Jenkins home page

The screenshot shows the Jenkins home page. At the top, there is a dark header with the Jenkins logo and a search bar. Below the header, there are several navigation links: 'Dashboard', 'New Item', 'Build History', 'Manage Jenkins', and 'My Views'. The main content area has a 'Welcome to Jenkins!' section with a message: 'This page is where your Jenkins jobs will be displayed. To get started, you can set up distributed builds or start building a software project.' Below that is a 'Start building your software project' section. Further down are sections for 'Set up a distributed build' and 'Learn more about distributed builds'.

Step 20: Now install docker in main machine

\$ apt install docker.io -y



```
Setting up runc (1.1.12-0ubuntu3) ...
Setting up dns-root-data (2023112702-willsync1) ...
Setting up bridge-utils (1.7.1-1ubuntu2) ...
Setting up pigz (2.8-1) ...
Setting up containerd (1.7.12-0ubuntu4) ...
Created symlink /etc/systemd/system/multi-user.target.wants/containerd.service → /usr/lib/systemd/system/containerd.service.
Setting up ubuntu-fan (0.12.16) ...
Created symlink /etc/systemd/system/multi-user.target.wants/ubuntu-fan.service → /usr/lib/systemd/system/ubuntu-fan.service.
Setting up docker.io (24.0.7-0ubuntu4) ...
info: Selecting GID from range 100 to 999 ...
info: Adding group `docker' (GID 114) ...
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /usr/lib/systemd/system/docker.service.
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket → /usr/lib/systemd/system/docker.socket.
Processing triggers for dbus (1.14.10-4ubuntu4) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

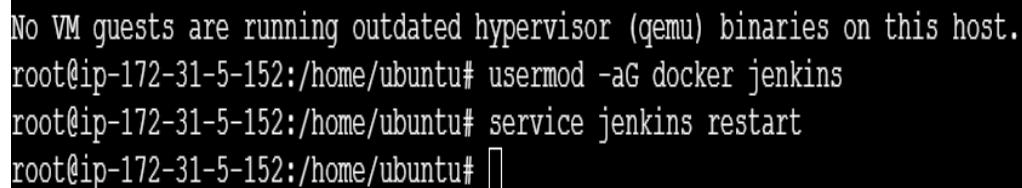
No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@ip-172-31-5-152:/home/ubuntu# 
```

i-0f5f05c6c2c216ee9 (Final-project-Mater)
PublicIPs: 52.66.253.128 PrivateIPs: 172.31.5.152

- Now execute below command to give permission to docker then restart the jenkins

\$usermod -aG docker jenkins

\$service jenkins restart

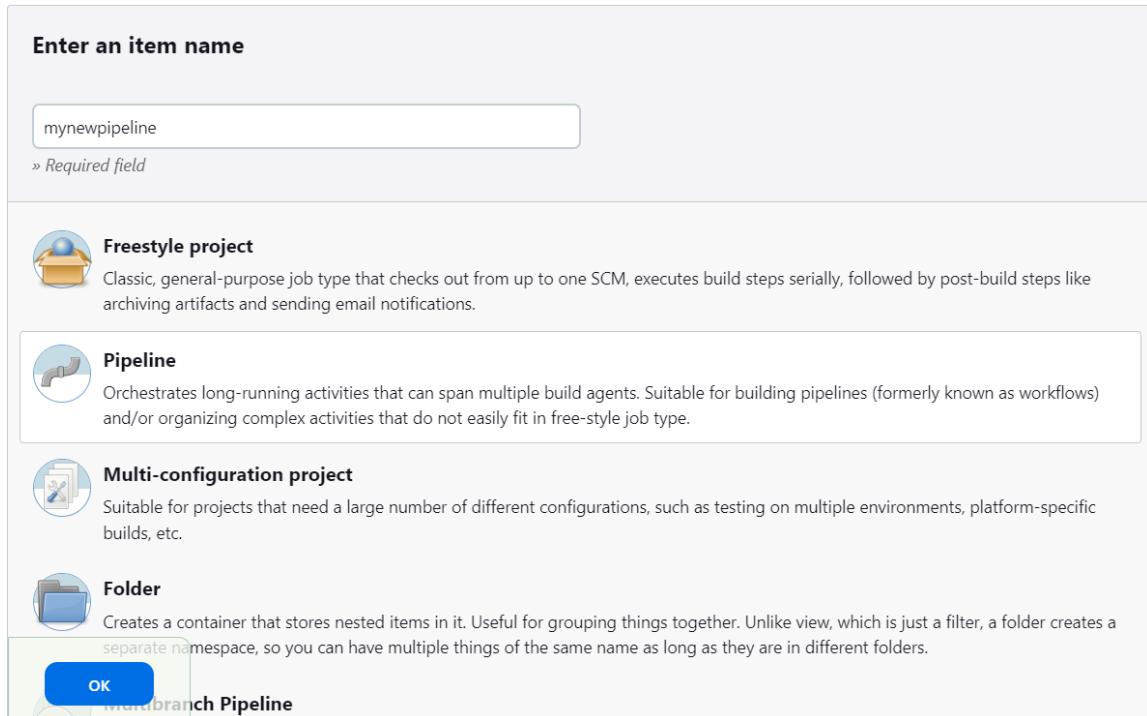


```
No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@ip-172-31-5-152:/home/ubuntu# usermod -aG docker jenkins
root@ip-172-31-5-152:/home/ubuntu# service jenkins restart
root@ip-172-31-5-152:/home/ubuntu# 
```

i-0f5f05c6c2c216ee9 (Final-project-Mater)

PublicIPs: 52.66.253.128 PrivateIPs: 172.31.5.152

Step 21: Now reload the Jenkins and click on new item, and create a pipeline with name newpipeline then click on ok



• Now click on pipeline

Not secure 52.66.253.128:8080/job/newpipeline/configure

Dashboard > newpipeline > Configuration

Advanced

Configure

General Advanced Project Options Pipeline

Definition

Pipeline script from SCM

SCM ?

None

Script Path ?

Jenkinsfile

Lightweight checkout ?

Pipeline Syntax

Save Apply

- Now select pipeline script from SCM by dropdown

The screenshot shows the Jenkins Pipeline Configuration page. At the top, there are several build triggers: 'Build periodically', 'GitHub hook trigger for GITScm polling', 'Poll SCM', 'Quiet period', and 'Trigger builds remotely (e.g., from scripts)'. Below these are sections for 'General', 'Advanced Project Options', and 'Pipeline'. Under 'Pipeline', the 'Definition' dropdown is set to 'Pipeline script from SCM'. At the bottom are 'Save' and 'Apply' buttons.

- Now select Git in SCM(source code management)

The screenshot shows the Jenkins Pipeline configuration under the 'Pipeline' section. The 'Definition' dropdown is set to 'Pipeline script from SCM'. The 'SCM' dropdown is set to 'Git'. The 'Script Path' field contains 'Jenkinsfile1'. A checkbox for 'Lightweight checkout' is checked. At the bottom are 'Save' and 'Apply' buttons.

Step 22: Now go to Github repository of Banking-java-project

The screenshot shows a GitHub repository page for 'sumitsingh231/Banking-java-project'. The repository is public and was forked from 'akshu20791/Banking-java-project'. The 'Code' tab is selected. The 'master' branch is shown with 1 branch and 0 tags. There are 18 commits in total. Recent commits include 'Create Jenkinsfile1' by sumitsingh231, 'first commit' for '.mvn/wrapper' and 'src', and 'Update application.properties' for 'src'. The repository has 0 stars, 0 forks, and 0 releases.

- There Create a new file with name **jenkinsfile1** and pasted the below code
- Code describe the below points
 - Taking the banking project
 - Start code compiling
 - Code testing
 - Package

- **Code-**

```

pipeline{
    agent any
    stages{
        stage('checkout the code from github'){
            steps{
                git url: 'https://github.com/sumitsingh231/Banking-java-project/'
                echo 'github url checkout'
            }
        }
        stage('codecompile with sumit'){
            steps{
                echo 'starting compiling'
                sh 'mvn compile'
            }
        }
        stage('codetesting with sumit'){
            steps{
                sh 'mvn test'
            }
        }
        stage('qa with sumit'){
    
```

```

steps{
    sh 'mvn checkstyle:checkstyle'

}

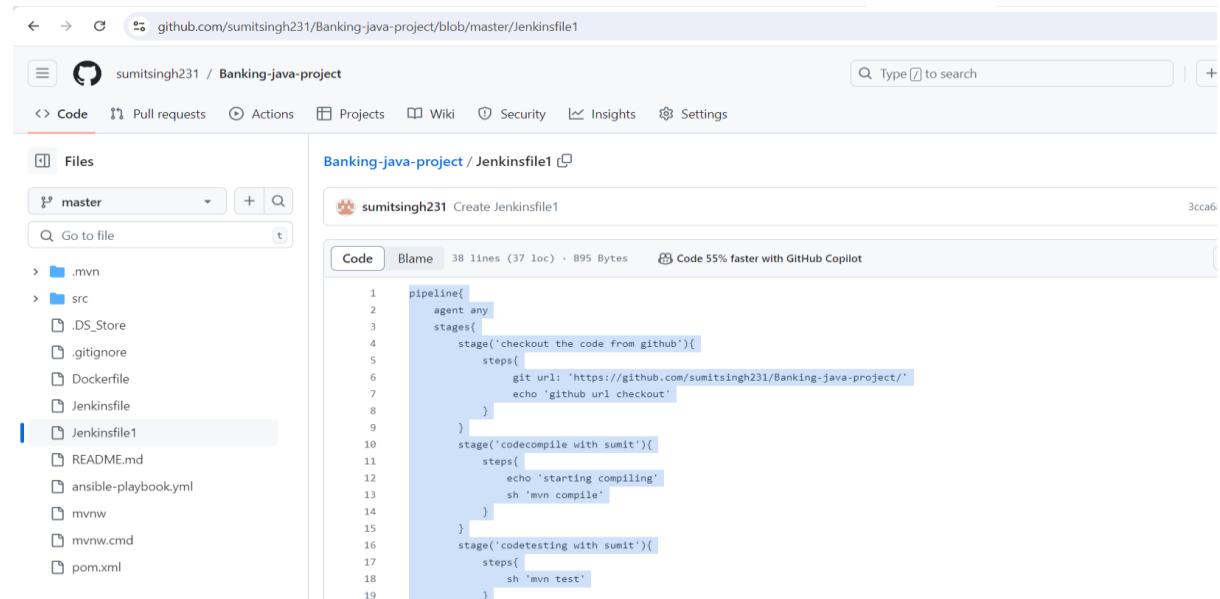
}

stage('package with sumit'){
    steps{
        sh 'mvn package'
    }
}

stage('run dockerfile'){
    steps{
        sh 'docker build -t myimg'
    }
}

}

```



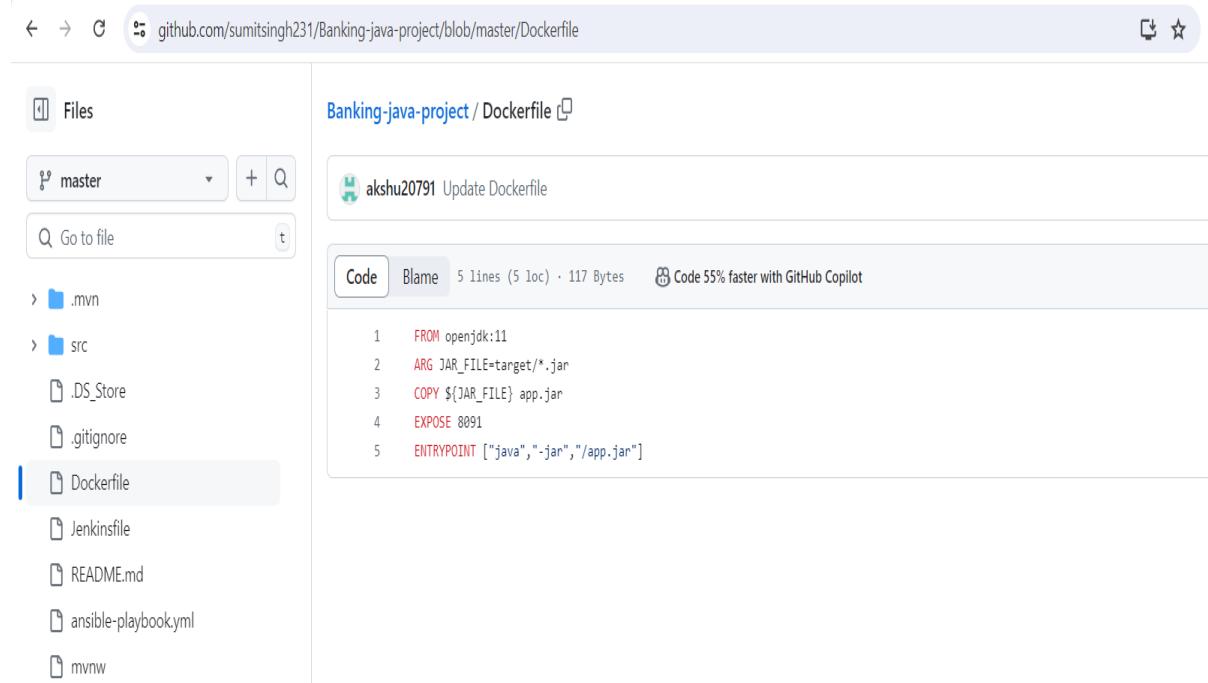
The screenshot shows a GitHub repository page for 'sumitsingh231 / Banking-java-project'. The 'Code' tab is selected, displaying the Jenkinsfile1 content. The file contains Jenkins pipeline code for a Java project, including stages for code checkout, compilation, and testing.

```

1 pipeline{
2     agent any
3     stages{
4         stage('checkout the code from github'){
5             steps{
6                 git url: 'https://github.com/sumitsingh231/Banking-java-project/'
7                 echo 'github url checkout'
8             }
9         }
10        stage('codecompile with sumit'){
11            steps{
12                echo 'starting compiling'
13                sh 'mvn compile'
14            }
15        }
16        stage('codetesting with sumit'){
17            steps{
18                sh 'mvn test'
19            }
20        }
21    }
22 }

```

Step 23: Now click on dockerfile in repository and edit export port 8091

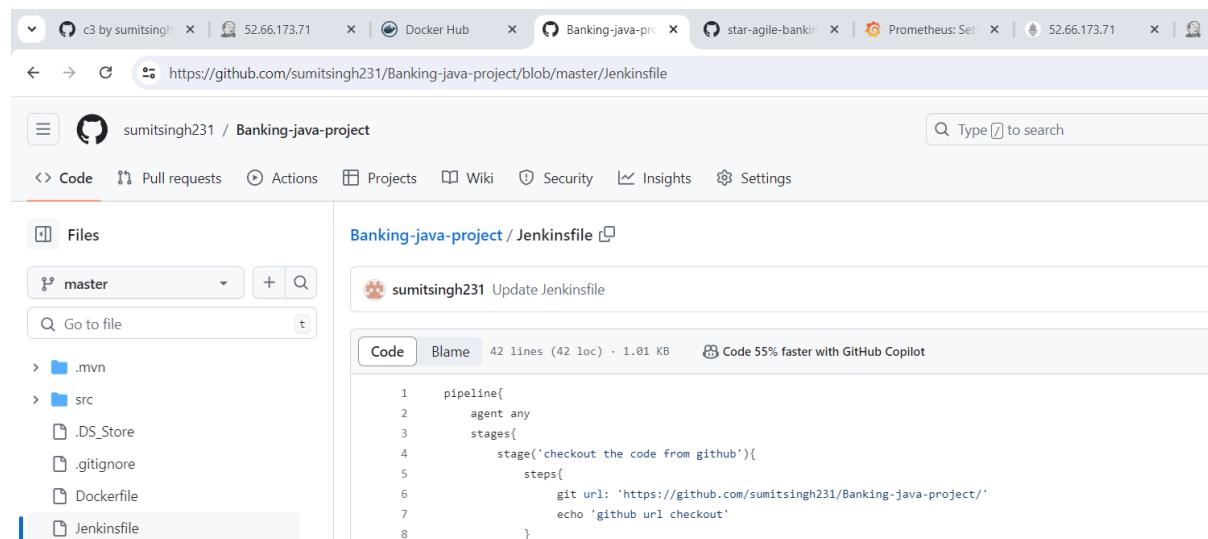


The screenshot shows a GitHub repository page for 'Banking-java-project'. The left sidebar lists files including '.mvn', 'src', '.DS_Store', '.gitignore', 'Dockerfile' (which is selected), 'Jenkinsfile', 'README.md', 'ansible-playbook.yml', and 'mvnw'. The main area displays the 'Dockerfile' content:

```
FROM openjdk:11
ARG JAR_FILE=target/*.jar
COPY ${JAR_FILE} app.jar
EXPOSE 8091
ENTRYPOINT ["java", "-jar", "/app.jar"]
```

Step 24: Copy the repository link

<https://github.com/sumitsingh231/Banking-java-project/>



The screenshot shows a GitHub repository page for 'Banking-java-project'. The left sidebar lists files including '.mvn', 'src', '.DS_Store', '.gitignore', 'Dockerfile', and 'Jenkinsfile' (which is selected). The main area displays the 'Jenkinsfile' content:

```
pipeline{
    agent any
    stages{
        stage('checkout the code from github'){
            steps{
                git url: 'https://github.com/sumitsingh231/Banking-java-project/'
                echo 'github url checkout'
            }
        }
    }
}
```

- Now come back to the Jenkins page and Paste this link into the Jenkins page

The screenshot shows the Jenkins Pipeline configuration screen for a job named 'newpipeline'. The left sidebar has tabs for General, Advanced Project Options, and Pipeline, with Pipeline selected. The main area is titled 'Pipeline script from SCM' and shows 'SCM ?' with 'Git' selected. Under 'Repositories ?', there is one entry for 'Repository URL ?' with the value 'https://github.com/sumitsingh231/Banking-java-project/'. Below it is a 'Credentials ?' section with a dropdown set to '- none -' and a '+ Add ▾' button. An 'Advanced ▾' button is also present. At the bottom are 'Add Repository' and 'Branches to build ?' buttons, and 'Save' and 'Apply' buttons at the bottom.

- Now click on save

The screenshot shows the Jenkins Pipeline Syntax configuration screen. It includes sections for '(Auto)', 'Additional Behaviours' (with an 'Add ▾' button), 'Script Path ?' containing 'Jenkinsfile1', and a checked checkbox for 'Lightweight checkout ?'. At the bottom are 'Save' and 'Apply' buttons.

Step 25: Now click on build now, You can see in build history build get executed successfully

The screenshot shows the Jenkins interface for a pipeline named "newpipeline". On the left, there's a sidebar with options like Status, Changes, Build Now, Configure, Delete Pipeline, Stages, Rename, and Pipeline Syntax. The main area has a green checkmark icon and the pipeline name "newpipeline". Below it, there's a section titled "Permalinks" with a list of recent builds. A large callout box highlights the "Build History" section, which shows a single entry for build #1, dated Aug 6, 2024, at 8:10 AM. It includes links for Atom feed for all and Atom feed for failures.

Status

newpipeline

Changes

Build Now

Configure

Delete Pipeline

Stages

Rename

Pipeline Syntax

Permalinks

- Last build (#1), 2 min 18 sec ago
- Last stable build (#1), 2 min 18 sec ago
- Last successful build (#1), 2 min 18 sec ago
- Last completed build (#1), 2 min 18 sec ago

Build History

trend

#1

Aug 6, 2024, 8:10 AM

Atom feed for all Atom feed for failures

- Now click on #1 (Build) to see the build

This screenshot shows the detailed view of build #1 from the previous history. It displays the same "Build History" header and "trend" dropdown. The main content area shows the single entry for build #1 on Aug 6, 2024, at 8:10 AM. At the bottom, there are two "Atom feed" links: "Atom feed for all" and "Atom feed for failures".

Build History

trend

#1

Aug 6, 2024, 8:10 AM

Atom feed for all Atom feed for failures

- Now click on pipeline console

The screenshot shows a Jenkins Pipeline Console for a job named 'newpipeline' with build #1. The build status is 'Success' and it completed 3 minutes 30 seconds ago. The pipeline stages listed on the left are: Checkout SCM, checkout the code from github, codecompile with sumit, codetesting with sumit, qa with sumit, package with sumit, and run dockerfile. The 'run dockerfile' stage is highlighted with a yellow background. The right side shows the details for the 'Stage 'run dockerfile'' which started 2 minutes 37 seconds ago, took 23 seconds, and was successful. It also includes a 'View as plain text' link. Below this, the console output for the 'docker build -t myimg .' command is shown, displaying steps 0 through 6.

```
0 + docker build -t myimg .
1 DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
2 Install the buildx component to build images with BuildKit:
3 https://docs.docker.com/go/buildx/
4
5 Sending build context to Docker daemon 58.66MB
6 Step 1/5 : FROM openjdk:11
```

- Here you can see the console output

- Now click on the pipeline overview, here you can see the execution in graph format

The screenshot shows the Jenkins Pipeline Overview for Build #1. At the top, there's a navigation bar with links for Dashboard, newpipeline, #1, Pipeline Overview, and a search bar. Below the navigation is a header for 'Build #1' with buttons for Rebuild, Console, and Configure. The main area is divided into two sections: 'Pipeline' and 'Details'. The 'Pipeline' section contains a horizontal timeline with nodes: Start, Checkout SCM, checkout the co..., codecompile wit..., codetesting with..., qa with sumit, package with su..., run dockerfile, and End. Each node has a green checkmark icon. To the right of the timeline is a 'Details' panel containing information: Manually run by sumit, Started 4 min 41 sec ago, Queued 22 ms, Took 1 min 17 sec.

- You can see I have build the image
- In above pic you can see the build steps
 - Start the build
 - Checkout the code from Github
 - Code compile with sumit
 - Code testing with sumit
 - QA with sumit

- Now renaming our instance as Master, Host and Monitoring

The screenshot shows the AWS CloudWatch Instances console. The top navigation bar includes 'Search [Alt+S]', 'Mumbai', and 'Submit'. Below the navigation is a search bar and a 'Launch instances' button. A table lists four instances: Mater, Monitoring, Host, and terraformmachine. The 'Host' instance is selected, indicated by a blue border around its row. The table columns include Name, Instance ID, Instance state, Instance type, Status check, Alarm status, and Availability Zone. All instances are listed as 'Running'.

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
Mater	i-0f5f05c6c2c216ee9	Running	t2.medium	2/2 checks passed	View alarms	ap-south-1b
Monitoring	i-00dc2969aefcc96b	Running	t2.medium	2/2 checks passed	View alarms	ap-south-1b
Host	i-04c252fe8cff36c7e	Running	t2.medium	2/2 checks passed	View alarms	ap-south-1b
terraformmachine	i-0b8e3124aec5c7928	Running	t3.medium	2/2 checks passed	View alarms	ap-south-1a

Step 26: Now install ansible inMaster instance

- **Browse install ansible on ubuntu and click on ansible documentation**

google.com/search?q=install+ansible+on+ubuntu&rlz=1C1ONGR_enIN1103IN1103&oq=install+ansible+on+ubuntu&gs_lcrp=EgZjaHJvI

install ansible on ubuntu

All Videos Images Shopping News Books Maps More Tools

Ansible Documentation
https://docs.ansible.com › latest › installation_distros

Installing Ansible on specific operating systems
Installing Ansible on Debian ; In the following example, we assume that you have wget and gpg already installed (sudo apt install wget gpg) ; Run the following ...

- [Installing Ansible on Fedora Linux](#)
- [Installing Ansible from EPEL](#)
- [Installing Ansible on OpenSUSE Tumbleweed/Leap](#)
- [Installing Ansible on Ubuntu](#)
- [Installing Ansible on Debian](#)
- [Installing Ansible on Arch Linux](#)
- [Installing Ansible on Windows](#)

- **Scroll down and click on Installing Ansible on ubuntu**

- [Installing Ansible on Fedora Linux](#)
- [Installing Ansible from EPEL](#)
- [Installing Ansible on OpenSUSE Tumbleweed/Leap](#)
- [Installing Ansible on Ubuntu](#)
- [Installing Ansible on Debian](#)
- [Installing Ansible on Arch Linux](#)
- [Installing Ansible on Windows](#)

- **Copy all commands**

sudo apt update

sudo apt install software-properties-common

sudo add-apt-repository --yes --update ppa:ansible/ansible

sudo apt install ansible

Installing Ansible on Ubuntu

Ubuntu builds are available [in a PPA here](#).

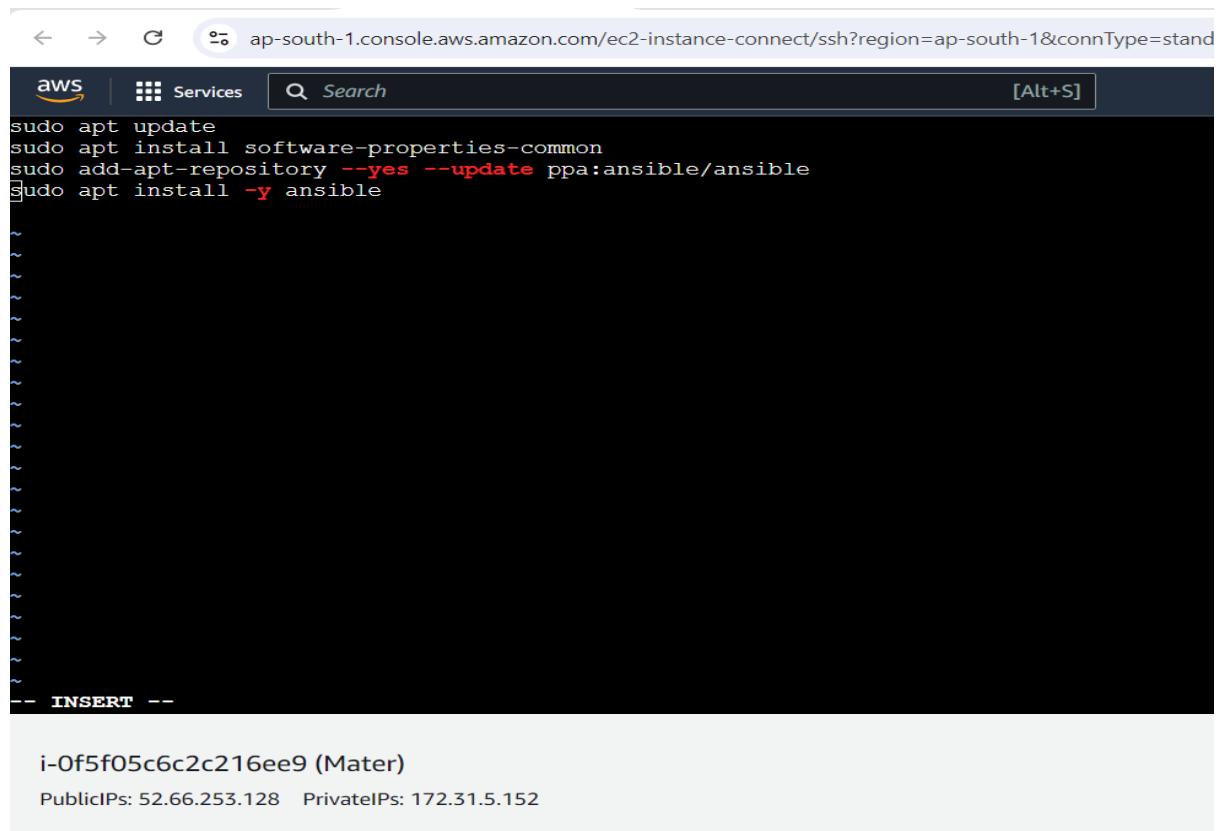
To configure the PPA on your system and install Ansible run these commands:

```
$ sudo apt update
$ sudo apt install software-properties-common
$ sudo add-apt-repository --yes --update ppa:ansible/ansible
$ sudo apt install ansible
```

Note

- **Create a file with name ansible.sh and paste copied command**

\$ vi ansible.sh



The screenshot shows a terminal window in the AWS Cloud9 IDE. The title bar indicates the session is connected to an EC2 instance in the ap-south-1 region. The terminal window displays the following commands:

```
sudo apt update
sudo apt install software-properties-common
sudo add-apt-repository --yes --update ppa:ansible/ansible
sudo apt install -y ansible
```

The terminal prompt shows several tilde (~) characters, indicating the user is in the root directory. At the bottom of the terminal window, it says "-- INSERT --". Below the terminal, the instance identifier is shown as i-0f5f05c6c2c216ee9 (Mater), and the public and private IP addresses are listed as 52.66.253.128 and 172.31.5.152 respectively.

Step 27: Now Execute below command to run script file

\$sh ansible.sh

The screenshot shows a terminal window within the AWS CloudShell interface. The URL at the top is ap-south-1.console.aws.amazon.com/ec2-instance-connect/ssh?region=ap-south-1&connType=standard&instanceId=i-0f5f05c6c2c216ee9&osUser=root. The terminal header includes the AWS logo, services menu, search bar, and Alt+S keybinding.

```
Setting up ansible-core (2.16.9-1ppa~noble) ...
Setting up python3-winrm (0.4.3-2) ...
Setting up ansible (9.8.0-1ppa~noble) ...
Setting up python3-paramiko (2.12.0-2ubuntu4) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.

root@ip-172-31-5-152:/home/ubuntu# ansible --version
ansible [core 2.16.9]
  config file = /etc/ansible/ansible.cfg
  configured module search path = ['/root/.ansible/plugins/modules', '/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/lib/python3/dist-packages/ansible
  ansible collection location = /root/.ansible/collections:/usr/share/ansible/collections
  executable location = /usr/bin/ansible
  python version = 3.12.3 (main, Apr 10 2024, 05:33:47) [GCC 13.2.0] (/usr/bin/python3)
  jinja version = 3.1.2
  libyaml = True
root@ip-172-31-5-152:/home/ubuntu# 
```

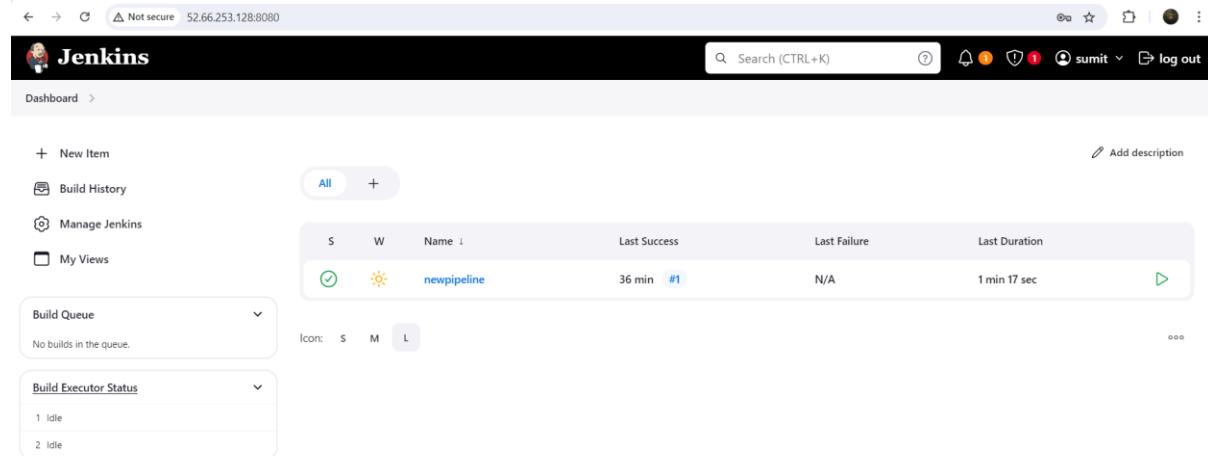
i-0f5f05c6c2c216ee9 (Mater)

PublicIPs: 52.66.253.128 PrivateIPs: 172.31.5.152

- You can see we have installed ansible in master machine

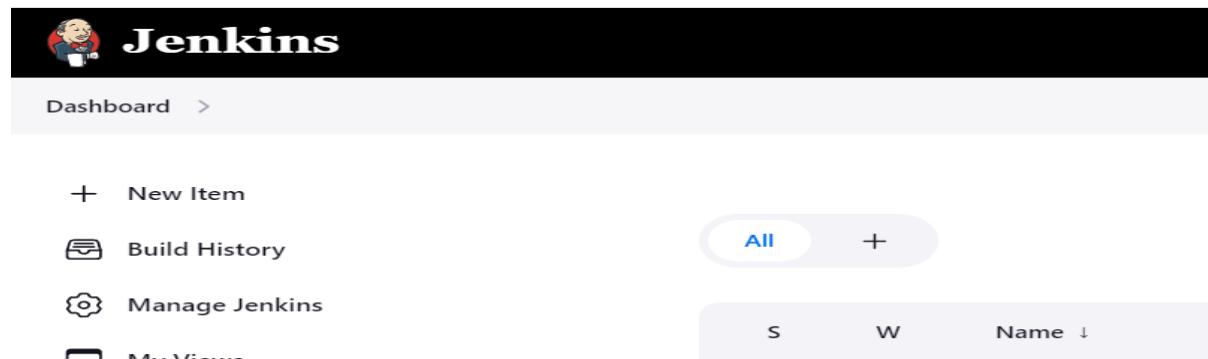
Step 28: Now gain restart the Jenkins and refresh the Jenkins page

\$service jenkins restart



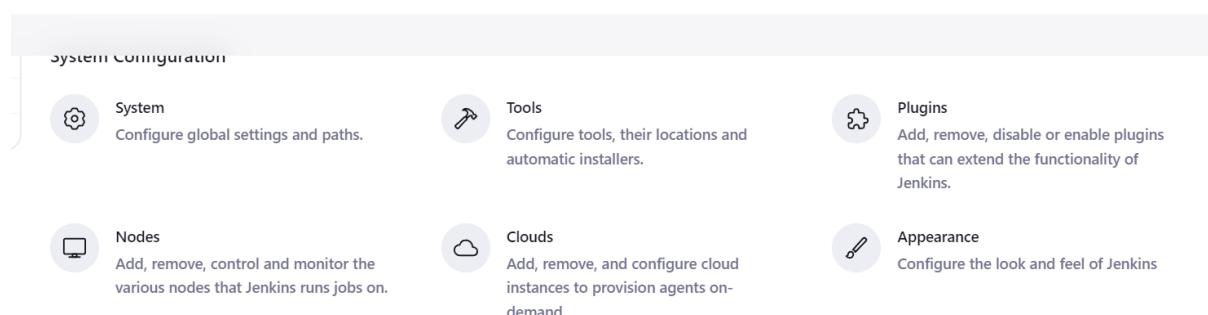
The screenshot shows the Jenkins dashboard at the URL 52.66.253.128:8080. The top navigation bar includes links for 'Dashboard', 'Build History', 'Manage Jenkins', and 'My Views'. A search bar and a 'log out' button are also present. The main content area displays a table of builds. One build, 'newpipeline', is listed with a green checkmark icon, indicating success. The table columns are labeled 'S' (Status), 'W' (Workstation), 'Name', 'Last Success', 'Last Failure', and 'Last Duration'. Below the table, there are sections for 'Build Queue' (empty) and 'Build Executor Status' (showing 1 idle and 2 idle executors).

- Click on manage Jenkins



The screenshot shows the 'Manage Jenkins' page. The top navigation bar includes links for 'Dashboard', 'Build History', 'Manage Jenkins', and 'My Views'. The main content area features a table with columns 'S', 'W', and 'Name'. On the left, there is a sidebar with links for 'New Item', 'Build History', 'Manage Jenkins', and 'My Views'. The 'Manage Jenkins' link is highlighted.

- Click on plugins



The screenshot shows the 'System Configuration' page. It lists several management options: 'System' (Configure global settings and paths), 'Tools' (Configure tools, their locations and automatic installers), 'Nodes' (Add, remove, control and monitor the various nodes that Jenkins runs jobs on), 'Clouds' (Add, remove, and configure cloud instances to provision agents on-demand), 'Plugins' (Add, remove, disable or enable plugins that can extend the functionality of Jenkins), and 'Appearance' (Configure the look and feel of Jenkins). Each option has a brief description and a corresponding icon.

- Click on Available plugins

The screenshot shows the Jenkins plugin manager interface. At the top, there's a navigation bar with links to 'Dashboard', 'Manage Jenkins', and 'Plugins'. Below this is a search bar labeled 'Search plugin updates'. On the left, a sidebar has four options: 'Updates' (selected), 'Available plugins' (highlighted in blue), 'Installed plugins', and 'Advanced settings'. The main content area is currently empty, indicating no available updates.

- Type ansible for search and select ansible plugin and click on install and select restart option

The screenshot shows the Jenkins plugin manager interface after searching for 'ansible'. The search results list two items: 'Ansible' and 'Ansible Tower'. The 'Ansible' plugin is selected, and its details are shown: it was released 1 month and 19 days ago. There is a blue 'Install' button next to the plugin name. The sidebar on the left remains the same as in the previous screenshot.

- You can see Jenkins restarted

The screenshot shows the Jenkins sign-in page. It features a large cartoon Jenkins character logo on the left and a sign-in form on the right. The form includes fields for 'Username' and 'Password', a 'Keep me signed in' checkbox, and a blue 'Sign in' button. The URL in the browser address bar is 52.66.253.128:8080/login?from=%2FpluginManager%2Fupdates%2F.

- Login again and click on installed plugin you will see there is ansible plugin which we have installed

The screenshot shows the Jenkins Plugin Manager interface. On the left, there's a sidebar with options: Updates, Available plugins, Installed plugins (which is selected), and Advanced settings. The main area has a search bar at the top. Below it, a table lists installed plugins. The columns are Name (sorted by name), Description, and Enabled (with a checkbox). The Ansible plugin is listed with its version (403.v8d0ca_dcb_b_502) and a note about invoking Ad-Hoc commands and playbooks. The Ant and Apache HttpComponents Client 4.x API plugins are also listed with their descriptions. The ASM API plugin is shown but not detailed.

Name	Enabled
Ansible plugin 403.v8d0ca_dcb_b_502 Invoke Ansible Ad-Hoc commands and playbooks. Report an issue with this plugin	<input checked="" type="checkbox"/> <input type="checkbox"/>
Ant Plugin 497.v94e7d9fffa_b_9 Adds Apache Ant support to Jenkins Report an issue with this plugin	<input checked="" type="checkbox"/> <input type="checkbox"/>
Apache HttpComponents Client 4.x API Plugin 4.5.14-208.v438351942757 Bundles Apache HttpComponents Client 4.x and allows it to be used by Jenkins plugins. Report an issue with this plugin	<input checked="" type="checkbox"/> <input type="checkbox"/>
ASM API Plugin 9.7-33.v4d23ef79fcc8	

- Now click on manage Jenkins and go to tools option for configure the ansible tool

The screenshot shows the Jenkins Manage Jenkins page. On the left, there's a sidebar with options: New Item, Build History, Manage Jenkins (selected), My Views, Build Queue (with a note: 'No builds in the queue.'), Build Executor Status (with notes: '1 Idle' and '2 Idle'), and System Configuration. The main area has a search bar at the top. Below it, a prominent message in red text says 'Java 11 end of life in Jenkins'. It states: 'You are running Jenkins on Java 11, support for which will end on or after Sep 30, 2024. Refer to [the documentation](#) for more details.' There are 'More Info' and 'Ignore' buttons. To the right, there are sections for System Configuration (with a note: 'Configure global settings and paths.'), Tools (with a note: 'Configure tools, their locations and automatic installers.'), and Plugins (with a note: 'Add, remove, disable or enable plugins that can extend the functionality of Jenkins.').

- Scroll down and click on add ansible

The screenshot shows the Jenkins interface at 52.66.253.128:8080/manage/configureTools/. The 'Tools' section is selected. Under 'Ansible installations', there is a form to 'Add Ansible'. The 'Name' field contains 'ansible' and the 'Install automatically' checkbox is checked. Below the form are 'Save' and 'Apply' buttons.

- Type name as ansible and tick on install automatically,then click first on apply then save

A detailed view of the 'Add Ansible' dialog box. It shows a 'Name' field with 'ansible' and an 'Install automatically' checkbox checked. There is also an 'Add Installer' dropdown menu. At the bottom are 'Save' and 'Apply' buttons.

- Now go to the dashboard

The screenshot shows the Jenkins dashboard at 52.66.253.128:8080/. The 'Dashboard' link is highlighted. A table lists the 'newpipeline' item with a green success icon, '51 min' last success, and 'N/A' last failure. Below the dashboard are sections for 'Build Queue' (empty) and 'Build Executor Status' (1 Idle).

- Click on newpipeline

The screenshot shows the Jenkins dashboard for the 'newpipeline' job. The job status is green with a checkmark, indicating it is successful. The pipeline has four stages: 'Changes', 'Build Now', 'Configure', and 'Delete Pipeline'. On the right, there is a 'Permalinks' section listing the last four builds, all of which are successful.

- Now go to configure and in pipeline script paste the code from Jenkins file

The screenshot shows the Jenkins configuration page for the 'newpipeline' job. The 'Pipeline' tab is selected. The 'Definition' dropdown is set to 'Pipeline script'. The 'Script' text area contains Groovy code for a multi-stage pipeline:

```

26  stage('package with sumit'){
27    steps{
28      sh 'mvn package'
29    }
30  }
31  stage('run dockerfile'){
32    steps{
33      sh 'docker build -t myimg .'
34    }
35  }
36  stage('port expose'){
37    steps{
38      sh 'docker run -dt -p 8091:8091 --name c001 myimg'
39    }
40  }
41 }
42 }

```

A checkbox for 'Use Groovy Sandbox' is checked. At the bottom, there are 'Save' and 'Apply' buttons.

- **Modify the code**

```
27
28     steps{
29         |   sh 'mvn package'
30     }
31     stage('run dockerfile'){
32         steps{
33             |   sh 'docker build -t sumitsingh231/myproject:1 .'
34         }
35     }
36     stage('Login the docker hub and push the file'){
37         steps{
38
39     }
40 }
41 }
```

- **Now click on pipeline syntax and select plugin**

Overview

This **Snippet Generator** will help you learn the Pipeline Script code which can be used to define various steps. Pick a step you are interested in from the list, configure it, click **Generate Pipeline Script**, and you will see a Pipeline Script statement that would call the step with that configuration. You may copy and paste the whole statement into your script or pick up just the options you care about. (Most parameters are optional and can be omitted in your script, leaving them at default values.)

Steps

Sample Step

withCredentials: Bind credentials to variables

withCredentials ?

Secret values are masked on a best-effort basis to prevent *accidental* disclosure. Multiline secrets, such as the contents of a SSH private key file, are not masked. See the inline help for details and usage guidelines.

- **Click on add then secret text,give name of variable as dockerpass,then click on add and select jenkins**

The screenshot shows the Jenkins Pipeline Snippet Generator interface. A 'Secret text' configuration panel is open, containing fields for 'Variable' (set to 'dockerhubpass') and 'Credentials' (a dropdown menu showing 'Jenkins' as an option). Below the panel is a 'Generate Pipeline Script' button.

- Select secret text in kind
- In secret giving dockerhub account password
- In ID typing dockerhubpassword
- In description dockerhubpass by sumit
- Then click on add

Jenkins Credentials Provider: Jenkins

Kind

Secret text

Scope

Global (Jenkins, nodes, items, all child items, etc)

Secret

.....

ID

dockerhubpasswd

Description

dockerpass by sumit singh

Cancel Add

- Now click on generate pipeline script, copy it

Generate Pipeline Script

```
withCredentials([string(credentialsId: 'dockerhubpassword', variable: 'dockerhubpass')]) {
    // some block
}
```

Global Variables

- Now go to pipeline syntax
- Type code as below,then click on apply then save

```

25
26      }
27      |     }
28      |   }
29      |
30      | }
31  } stage('run dockerfile'){
32  | steps{
33  |   sh 'docker build -t sumitsingh231/myproject:1 .'
34  | }
35  }
36  } stage('Login the docker hub and push the file'){
37  | steps{
38  |   withCredentials([string(credentialsId: 'dockerhubpassword', variable: 'dockerhubpass')]) {
39  |     sh 'docker login -u sumitsingh231 -p ${dockerhubpass}'.
40  }
41  |   }
42  }
43  }
44 }
```

Use Groovy Sandbox ?

Pipeline Syntax

[Save](#) [Apply](#)

- Now click on build now

← → ⚡ Not secure 52.66.253.128:8080/job/newpipeline/

Jenkins Search (CTF)

Dashboard > newpipeline >

[Status](#) [newpipeline](#)

</> Changes

▷ Build Now

⚙ Configure

Delete Pipeline

☰ Stages

✍ Rename

ⓘ Pipeline Syntax

Permalinks

- Last build (#1), 1 hr 43 min ago
- Last stable build (#1), 1 hr 43 min ago
- Last successful build (#1), 1 hr 43 min ago
- Last completed build (#1), 1 hr 43 min ago

Build History [trend](#) [/](#)

#1 | Aug 6, 2024, 8:10 AM

[Atom feed for all](#) [Atom feed for failures](#)

- You can see build success

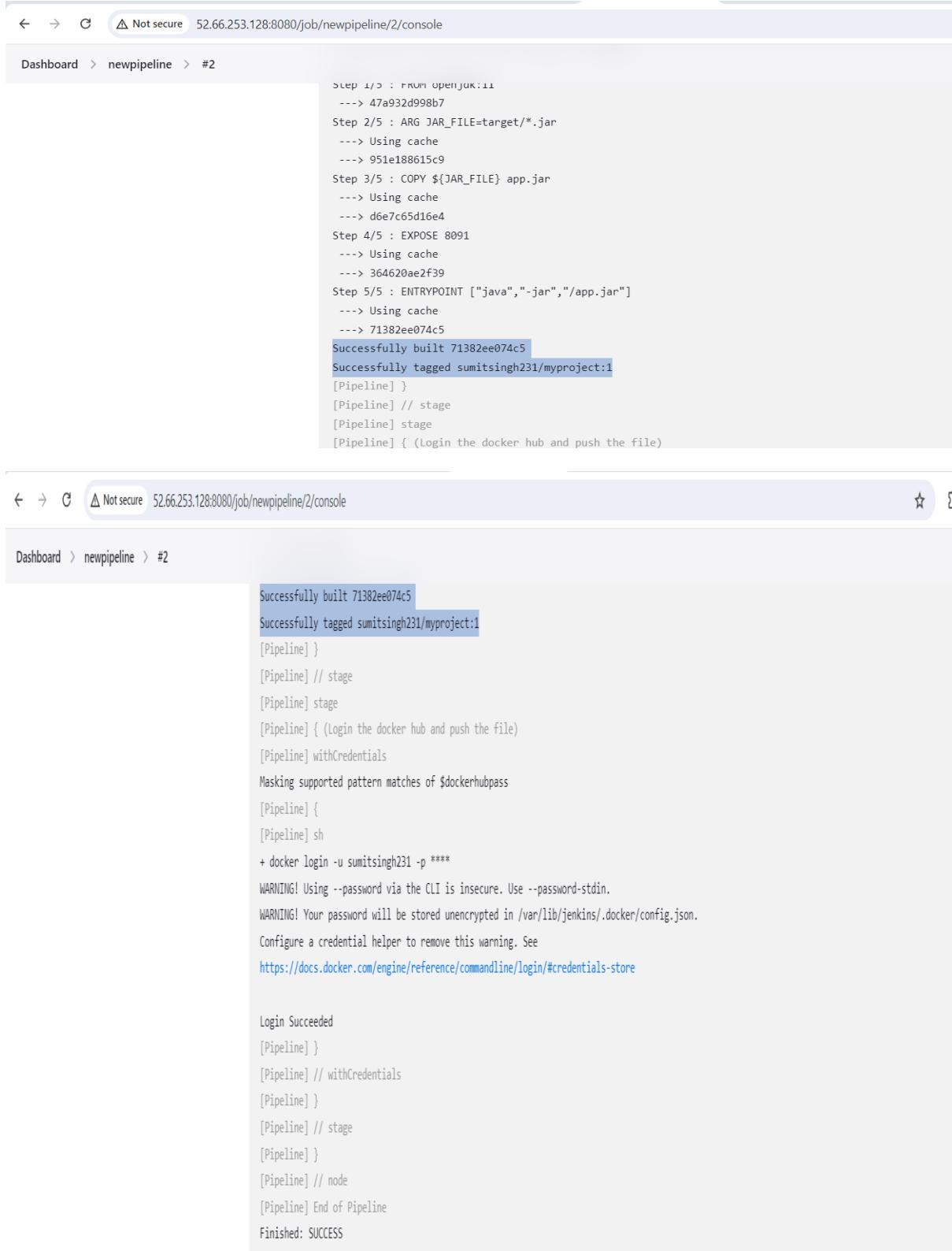
- Go to the pipeline console, you can see it will not show the password

```

Stage 'Login the docker hub and push the file'
Started 2 min 2 sec ago
Queued 0 ms
Took 2.4 sec
Success
View as plain text

dockersingh231@dockersingh231:~/jenkins/jenkins-slave$ docker login -u sumitsingh231 -p $(dockerhubpass)
WARNING! Using --password via the CLI is insecure. Use --password-stdin.
WARNING! Your password will be stored unencrypted in /var/lib/jenkins/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store
Login Succeeded
    
```

- In console output you can see it has build as well as tagged successfully



The screenshot shows two consecutive screenshots of a Jenkins pipeline console output. Both screenshots are from the same job, step #2, and show the same sequence of commands being executed.

```

Step 1/5 : FROM openjdk:11
--> 47a932d998b7
Step 2/5 : ARG JAR_FILE=target/*.jar
--> Using cache
--> 951e188615c9
Step 3/5 : COPY ${JAR_FILE} app.jar
--> Using cache
--> d6e7c65d16e4
Step 4/5 : EXPOSE 8091
--> Using cache
--> 364620ae2f39
Step 5/5 : ENTRYPOINT ["java","-jar","/app.jar"]
--> Using cache
--> 71382ee074c5
Successfully built 71382ee074c5
Successfully tagged sumitsingh231/myproject:1
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Login the docker hub and push the file)

```



```

Successfully built 71382ee074c5
Successfully tagged sumitsingh231/myproject:1
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Login the docker hub and push the file)
[Pipeline] withCredentials
Masking supported pattern matches of $dockerhubpass
[Pipeline] {
[Pipeline] sh
+ docker login -u sumitsingh231 -p ****
WARNING! Using -password via the CLI is insecure. Use --password-stdin.
WARNING! Your password will be stored unencrypted in /var/lib/jenkins/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
[Pipeline] }
[Pipeline] // withCredentials
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS

```

- Now again go to the configure and modify the file for push the image to docker hub,then click on apply then save

```

24      |     sh 'mvn checkstyle:checkstyle'
25    }
26  }
27  stage('package with sumit'){
28    steps{
29      |     sh 'mvn package'
30    }
31  }
32  stage('run dockerfile'){
33    steps{
34      |     sh 'docker build -t sumitsingh231/myproject:1 .'
35    }
36  }
37  stage('Login the docker hub and push the file'){
38    steps{
39      |     withCredentials([string(credentialsId: 'dockerhubpassword', variable: 'dockerhubpass')]) {
40        |       sh 'docker login -u sumitsingh231 -p ${dockerhubpass}'
41      }
42    }
43  }
44  stage('push to docker hub'){
45    steps{
46      |     sh 'docker push sumitsingh231/myproject:1'
47    }
48  }
49

```

Use Groovy Sandbox ?

Pipeline Syntax

Save **Apply**

- Now again click on build now,you can see build is success in console output

← → ⚙ Not secure 52.66.253.128:8080/job/newpipeline/3/console

Dashboard > newpipeline > #3

```

7b7f3078e1db: Preparing
826c3ddb29c: Preparing
b626401ef603: Preparing
9b55156abf26: Preparing
293d5db30c9f: Preparing
03127cdb479b: Preparing
9c742cd6c7a5: Preparing
293d5db30c9f: Waiting
03127cdb479b: Waiting
9c742cd6c7a5: Waiting
b626401ef603: Mounted from library/openjdk
826c3ddb29c: Mounted from library/openjdk
7b7f3078e1db: Mounted from library/openjdk
9b55156abf26: Mounted from library/openjdk
293d5db30c9f: Mounted from library/openjdk
03127cdb479b: Mounted from library/openjdk
9c742cd6c7a5: Mounted from library/openjdk
af257ecf62ce: Pushed
1: digest: sha256:24364f75e8e85daab6c2727c18a146a920462dfaab2e7e67e0690fe55a4ac9dc size: 2007
[Pipeline]
[Pipeline] // stage
[Pipeline]
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS

```

- Here also you can see build got success

The screenshot shows the Jenkins dashboard for a pipeline named "newpipeline". The status is green with a checkmark, indicating success. On the left, there's a sidebar with options like Status, Changes, Build Now, Configure, Delete Pipeline, Stages, Rename, and Pipeline Syntax. Below the sidebar is a "Build History" section with two entries: #3 (Aug 6, 2024, 10:22 AM) and #2 (Aug 6, 2024, 9:55 AM). To the right, under "Permalinks", there's a list of recent builds: Last build (#1), 1 hr 35 min ago; Last stable build (#1), 1 hr 35 min ago; Last successful build (#1), 1 hr 35 min ago; and Last completed build (#1), 1 hr 35 min ago.

Step 29: Now go to docker hub account and verify image has been pushed

- You can see image myproject:1 has been pushed successfully
- Last push 2 min ago

The screenshot shows the Docker Hub interface for the user "sumitsingh231". The "Repositories" tab is selected. Three repositories are listed: "sumitsingh231 / myproject" (last pushed 2 minutes ago), "sumitsingh231 / endtoendproject25may" (last pushed 8 days ago), and "sumitsingh231 / addsumit" (last pushed 27 days ago). On the right side, there's a sidebar with icons for creating a repository, organization, and user management, along with a link to "Create An Organization". Below the sidebar, there's a note about managing users and access.

- Click on myproject
- Here you can see the tag as 1

The screenshot shows a Docker Hub repository page. At the top, there is a navigation bar with links for 'Explore', 'Repositories' (which is underlined), and 'Organizations'. A search bar on the right contains the placeholder 'Search Docker'. Below the navigation bar, the repository path 'sumitsingh231 / [Repositories](#) / [myproject](#) / [General](#)' is displayed. A horizontal menu below the path includes 'General' (underlined), 'Tags', 'Builds', 'Collaborators', 'Webhooks', and 'Settings'. The main content area displays the repository details: 'sumitsingh231/myproject' with 1 tag, last updated 5 minutes ago. It notes that the repository does not have a description (labeled 'INCOMPLETE') or a category (also labeled 'INCOMPLETE'). A 'Tags' section below shows one tag: '1' (Image type, pushed 5 minutes ago). A link 'See all' is at the bottom of the tags section.

← → C hub.docker.com/repository/docker/sumitsingh231/myproject/general

dockerhub Explore **Repositories** Organizations Search Docker

sumitsingh231 / [Repositories](#) / [myproject](#) / [General](#)

General Tags Builds Collaborators Webhooks Settings

sumitsingh231/myproject 1

Updated 5 minutes ago

This repository does not have a description INCOMPLETE

This repository does not have a category INCOMPLETE

Tags

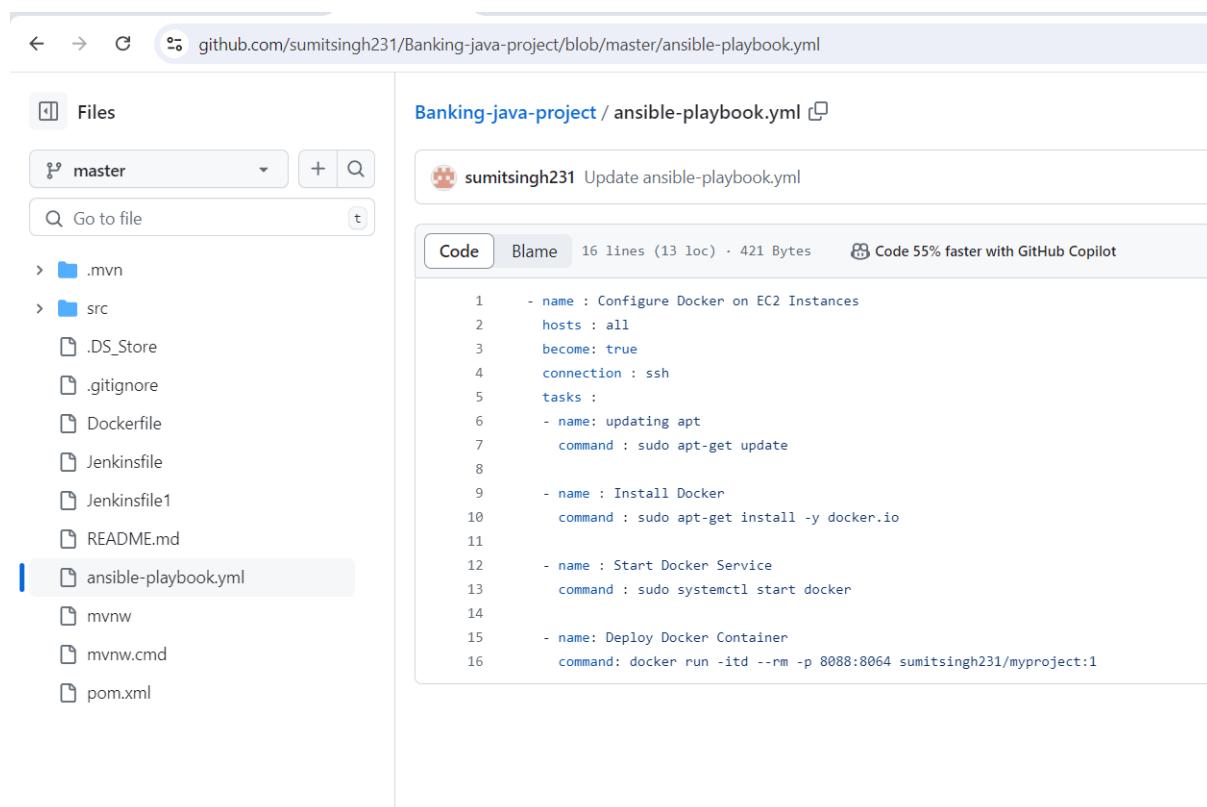
This repository contains 1 tag(s).

Tag	OS	Type	Pulled	Pushed
1		Image	4 minutes ago	5 minutes ago

[See all](#)

- Now going to Github repository of Banking project and there
- Click on ansible-playbook.yml file
-
- Modify the content as below

\$command: docker run -itd --rm -p 8088:8064
sumitsingh231/myproject:1



The screenshot shows a GitHub repository page for 'Banking-java-project'. The 'ansible-playbook.yml' file is selected in the sidebar. The main area displays the YAML code for the playbook:

```

1   - name : Configure Docker on EC2 Instances
2     hosts : all
3     become: true
4     connection : ssh
5     tasks :
6       - name: updating apt
7         command : sudo apt-get update
8
9       - name : Install Docker
10      command : sudo apt-get install -y docker.io
11
12      - name : Start Docker Service
13        command : sudo systemctl start docker
14
15      - name: Deploy Docker Container
16        command: docker run -itd --rm -p 8088:8064 sumitsingh231/myproject:1

```

Step 30: Now go to master instance and execute below command for define host

\$cd /etc/ansible/

```

root@ip-172-31-5-152:/home/ubuntu# service jenkins restart
root@ip-172-31-5-152:/home/ubuntu# cd /etc/ansible/
root@ip-172-31-5-152:/etc/ansible# ls
ansible.cfg  hosts  roles
root@ip-172-31-5-152:/etc/ansible# 

```

i-0f5f05c6c2c216ee9 (Mater)

PublicIPs: 52.66.253.128 PrivateIPs: 172.31.5.152

- Now execute below command

\$vi hosts

- Come to bottom and type

[demo-sumit]

Publicip of host server

The screenshot shows the AWS CloudWatch Metrics Insights interface. At the top, there's a search bar and various filter options. Below it is a table titled "Instances (1/4) Info" with columns for Name, Instance ID, Instance state, Instance type, Status check, Alarm status, and Availability. One row is selected, showing "Host" with Instance ID "i-04c252fe8cff36c7e". The bottom half of the screen shows a detailed view for this specific instance, with tabs for Details, Status and alarms, Monitoring, Security, Networking, Storage, and Tags. Under the Details tab, there's an "Instance summary" section with "Info" and "Public IPv4 address copied" (with a value of "65.0.199.19"). There are also sections for Instance ID ("i-04c252fe8cff36c7e (Host)"), IPv4 address, and Instance state ("Running"). On the right side, there are sections for "Private IPv4 addresses" (with "172.31.7.75") and "Public IPv4 DNS" (with "ip-2-66-0-199-19.compute.amazonaws.com").

[demo-sumit]

65.0.199.19

The screenshot shows a terminal window titled 'ap-south-1.console.aws.amazon.com/ec2-instance-connect/ssh?region=ap-south-1&connType=star'. The terminal displays an Ansible playbook configuration:

```

## www[001:006].example.com

# You can also use ranges for multiple hosts:
## db-[99:101]-node.example.com

# Ex 3: A collection of database servers in the 'dbservers' group:
## [dbservers]
## db01.intranet.mydomain.net
## db02.intranet.mydomain.net
## 10.25.1.56
## 10.25.1.57

# Ex4: Multiple hosts arranged into groups such as 'Debian' and 'openSUSE':
## [Debian]
## alpha.example.org
## beta.example.org

## [openSUSE]
## green.example.com
## blue.example.com
[demo-sumit]
65.0.199.19
-- INSERT --

```

Below the terminal, the instance details are shown:

i-0f5f05c6c2c216ee9 (Mater)
Public IPs: 52.66.253.128 Private IPs: 172.31.5.152

Step 31: Now come back to the pipeline

- click on pipeline syntax
- Select ansiblePlaybook:Invoke an ansible playbook
- Copy playbook file name

The screenshot shows a GitHub repository interface for 'Banking-java-project'. The file 'ansible-playbook.yml' is displayed. The code content is:

```

- name : Configure Docker on EC2 Instances
  hosts : all
  become: true
  connection : ssh

```

- Paste in playbook file path in workspace

Overview

This **Snippet Generator** will help you learn the Pipeline Script code which can be used to define various steps. Pick a step you are interested in from the list, configure it, click **Generate Pipeline Script**, and you will see a Pipeline Script statement that would call the step with that configuration. You may copy and paste the whole statement into your or pick up just the options you care about. (Most parameters are optional and can be omitted in your script, leaving them at default values.)

Steps

Sample Step

ansiblePlaybook: Invoke an ansible playbook

ansiblePlaybook ?

Ansible tool

ansible

Playbook file path in workspace

ansible-playbook.yml

Inventory file path in workspace

- In **Inventory file path type /etc/ansible/hosts**

Ansible tool

ansible

Playbook file path in workspace

ansible-playbook.yml

Inventory file path in workspace

/etc/ansible/hosts

- In **ssh connection credentials,click on add and select Jenkins**

SSH connection credentials

- none -

+ Add ▾

Jenkins

Vault credentials

- **Select SSH connection with private key**
- **In ID type ansible**
- **In description type ansible key file for Final project (financeme)**
- **In username type ubuntu**

Jenkins Credentials Provider: Jenkins

Global credentials (unrestricted)

Kind

SSH Username with private key



Scope ?

Global (Jenkins, nodes, items, all child items, etc)



ID ?

ansible

Description ?

ansible key file for final project financeme

Username

ubuntu

- **Click on Enter directly**
- **Now copy the key file of host I have used project1.pem key file so copying data of this key file and paste it into Key section then click on add**

Jenkins Credentials Provider: Jenkins

Treat username as secret ?

Private Key

Enter directly

Key

```
-----BEGIN RSA PRIVATE KEY-----
26r+zOMj0XZ9gEa1yyAEVsR+ur9gqDLKJM/g7AWZVhJVCJ60PRRTJ98joypW0P6
TRVTJxmKc0PAI+lHK9Vjs8piMbB6xxKo3PqL04h6+bQ4poBiSNog==
-----END RSA PRIVATE KEY-----
```

Passphrase

Enter New Secret Below

Cancel Add

- Now in SSH connection credentials select which we have created

Ansible tool

ansible

Playbook file path in workspace

ansible-playbook.yml

Inventory file path in workspace

/etc/ansible/hosts

SSH connection credentials

ubuntu (ansible key file for final project financeme)

+ Add ▾

Vault credentials

- none -

+ Add ▾

- Scroll down and checkout Use become

A screenshot of a web-based configuration interface for an Ansible playbook. At the top left is a button labeled '+ Add ▾'. Below it is a section titled 'Vault tmp path' with a text input field. Underneath are two checkboxes: 'Check mode' (unchecked) and 'Use become' (checked). A 'Become username' input field is also present.

- **Become username type root and sudo username keep blank**

A screenshot of the same configuration interface. The 'Use become' checkbox is checked. In the 'Become username' field, the value 'root' is entered. Below this, another checkbox 'Use sudo (deprecated)' is unchecked. A 'Sudo username (deprecated)' input field is also present.

- **Checkout the Disable the host SSH key check,then click on Generate pipeline script**

A screenshot of the configuration interface. The 'Disable the host SSH key check' checkbox is checked. Another checkbox 'Colorized output' is unchecked. A 'Extra parameters' input field is present. At the bottom is a blue button labeled 'Generate Pipeline Script'.

- Pipeline script is generated, copy it

The screenshot shows a user interface for generating a pipeline script. At the top, there is a blue button labeled "Generate Pipeline Script". Below it is a code editor containing the following Groovy script:

```
ansiblePlaybook become: true, credentialsId: 'ansible', disableHostKeyChecking: true, installation: 'ansible', inventory: '/etc/ansible/hosts', playbook: 'ansible-playbook.yml',  
sudoUser: null, vaultTmpPath: ''
```

Step 32: Now go to the pipeline and paste the script into line number 50, then click on apply then save

The screenshot shows a Groovy script editor with the following content:

```
40 }  
41 }  
42 }  
43 * stage('push to docker hub'){  
44 *   steps{  
45     sh 'docker push sumitsingh231/myproject:1'  
46   }  
47 }  
48 * stage('Deployment stage using ansible'){  
49 *   steps{  
50     ansiblePlaybook become: true, credentialsId: 'ansible', disableHostKeyChecking: true, installation: 'ansible', inventory:  
51     ''  
52   }  
53 }  
54 }
```

The line 50 contains the generated Ansible playbook configuration.



Pipeline Syntax

Save

Apply

Step 33: We have restarted Our Mater, Host , and Monitor machine so public ip has changed

- **Master Machine-**

Public ip: 13.201.75.139

```
ubuntu@ip-172-31-5-152:~$ ls
ansible.sh  jenkins.sh  node_exporter-1.8.2.linux-amd64  node_exporter-1.8.2.linux-amd64.tar.gz
ubuntu@ip-172-31-5-152:~$
```

i-0f5f05c6c2c216ee9 (Master)

PublicIPs: 13.201.75.139 PrivateIPs: 172.31.5.152

- **Host machine-**

Public ip: 65.0.27.52

```
node_exporter-1.8.2.linux-amd64  node_exporter-1.8.2.linux-amd64.
root@ip-172-31-7-75:/home/ubuntu# docker ps -a
CONTAINER ID        IMAGE               COMMAND
49c0a61fcf96      sumitsingh231/myproject:1   "java -jar /app.jar"
root@ip-172-31-7-75:/home/ubuntu#
```

i-04c252fe8cff36c7e (Host)

PublicIPs: 65.0.27.52 PrivateIPs: 172.31.7.75

- **Monitoring Machine-**

Public ip: 13.233.153.5

```
ts=2024-08-07T18:02:05.902Z caller=main.go:1415 level=info
ge=0s remote_storage=1.762µs web_handler=0s query_engine
5µs
ts=2024-08-07T18:02:05.902Z caller=main.go:1145 level=info
ts=2024-08-07T18:02:05.902Z caller=manager.go:164 level=
[]
```

i-00dcb2969aefcc96b (Monitoring)

PublicIPs: 13.233.153.5 PrivateIPs: 172.31.15.199

Step 34: Now again click on build now,you can see #24

Successfully build

← → ⌂ Not secure 13.201.75.139:8080/job/newpipeline/

Jenkins

Dashboard > newpipeline >

Status newpipeline

</> Changes
▷ Build Now
⚙ Configure
Delete Pipeline
☷ Stages
✍ Rename
ℹ Pipeline Syntax

Permalinks

- Last build (#24), 1 hr 19 min ago
- Last stable build (#24), 1 hr 19 min ago
- Last successful build (#24), 1 hr 19 min ago
- Last failed build (#21), 1 hr 38 min ago
- Last unsuccessful build (#21), 1 hr 38 min ago
- Last completed build (#24), 1 hr 19 min ago

Build History trend Filter... /

- #24 | Aug 7, 2024, 5:12 PM
- #23 | Aug 7, 2024, 5:07 PM

- **Build history-**

Build History trend Filter... /

- #24 | Aug 7, 2024, 5:12 PM
- #23 | Aug 7, 2024, 5:07 PM
- #22 | Aug 7, 2024, 4:57 PM
- (✗) #21

❖ In console output build is successfully finished

← → ⌂ Not secure 13.201.75.139:8080/job/newpipeline/24/console

Dashboard > newpipeline > #24

```

TASK [Gathering Facts] ****
ok: [65.0.27.52]

TASK [updating apt] ****
changed: [65.0.27.52]

TASK [Install Docker] ****
changed: [65.0.27.52]

TASK [Start Docker Service] ****
changed: [65.0.27.52]

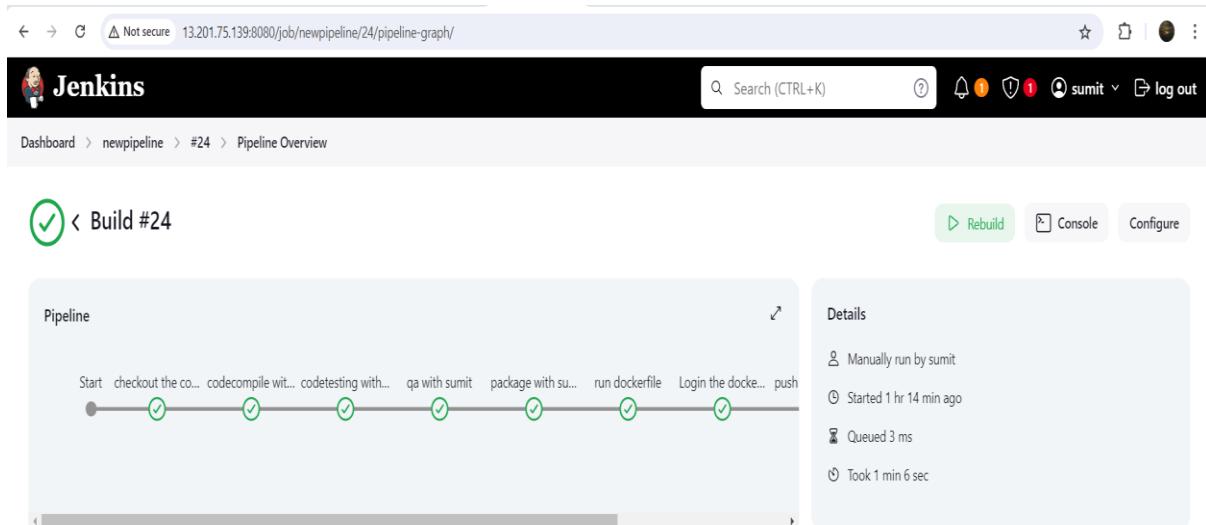
TASK [Deploy Docker Container] ****
changed: [65.0.27.52]

PLAY RECAP ****
65.0.27.52 : ok=5    changed=4    unreachable=0    failed=0    skipped=0    rescued=0

[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS

```

❖ You can also see in pipeline overview



- Expand View-



❖ Now you also can see the pipeline console-

A screenshot of the Jenkins Pipeline Console for Build #24. The URL is 13.201.75.139:8080/job/newpipeline/24/pipeline-console/. The page shows the build status as Success 1 hr 16 min ago in 1 min 6 sec. On the left, a list of stages is shown with green checkmarks: checkout the code from github, codecompile with sumit, codetesting with sumit, qa with sumit, package with sumit, run dockerfile, Login the docker hub and push the file, push to docker hub, and Deployment stage using ansible. The last stage is highlighted with a yellow background. To the right, the 'Deployment stage using ansible' details are displayed, including start time, queue time, duration, success status, and a link to view the logs as plain text. Below this, a section titled 'Invoke an ansible playbook' shows the command history:

```

1 [WARNING]: Invalid characters were found in group names due
2 -vvvv to see details
3
4 PLAY [Configure Docker on EC2 Instances] *****
5
6 TASK [Gathering Facts] *****
7 ok: [65.0.27.52]
8
9 TASK [updating apt] *****

```

Step 35: Now connect the host server ans execute below command in this

\$docker ps

\$docker images

```
root@ip-172-31-7-75:/home/ubuntu# docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
49c0a61fcf96 sumitsingh231/myproject:1 "java -jar /app.jar" About an hour ago Up About an hour 0.0.0:8088->8064/tcp, :::8088->8064/tcp modest_bose
root@ip-172-31-7-75:/home/ubuntu# docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
sumitsingh231/myproject 1 ce22249cf0db 27 hours ago 696MB
root@ip-172-31-7-75:/home/ubuntu# ]
```

i-04c252fe8cff36c7e (Host)

PublicIPs: 65.0.27.52 PrivateIPs: 172.31.7.75

- cropped image-**

```
root@ip-172-31-7-75:/home/ubuntu# docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS
49c0a61fcf96 sumitsingh231/myproject:1 "java -jar /app.jar" About an hour ago Up About an hour 0.0.0:8088->8064/tcp,
root@ip-172-31-7-75:/home/ubuntu# docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
sumitsingh231/myproject 1 ce22249cf0db 27 hours ago 696MB
root@ip-172-31-7-75:/home/ubuntu# ]
```

i-04c252fe8cff36c7e (Host)

PublicIPs: 65.0.27.52 PrivateIPs: 172.31.7.75

- Now by using below ip (Host Machine public ip)and port 8088 we can access the project**

`docker build -t sumitsingh231/myproject:1 .`

```
PLAY RECAP ****
65.0.27.52 : ok=5    changed=4    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

`docker build -t sumitsingh231/myproject:1 .`

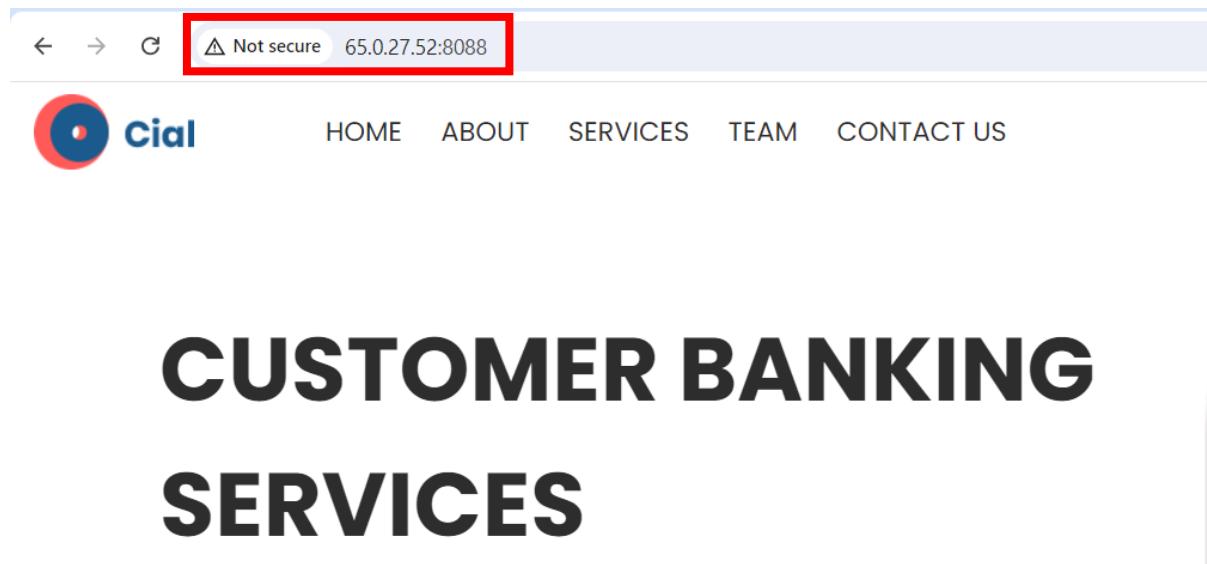
Step 36: Now browse public ip of host machine with port 8088

http:// 65.0.27.52:8088



You can see we can access our banking project(financeme)

Cropped image-



- ❖ Hence we accessed it on browser by using Host machine ip address

Step 37: Now installing Prometheus in Monitoring machine

- Connect the instance and execute below command
`$sudo su`
- Now create a file(prom.sh) by execute below command
`$vi prom.sh`
- Now browse install Prometheus on ubuntu on browser
- Open official website
- Click on download

The screenshot shows a web browser window with multiple tabs open. The active tab is 'Download | Prometheus' at prometheus.io/download/. The page displays a sidebar with exporter options: mysql_exporter, node_exporter, promlens, pushgateway, and statsd_exporter. Below this, there are dropdown menus for 'Operating system' set to 'popular' and 'Architecture' set to 'amd64'. A main table lists download links for Prometheus releases:

File name	OS	Arch	Size	SHA256 Checksum
prometheus-2.54.0-rc.0.darwin-amd64.tar.gz	darwin	amd64	101.25 MiB	0f3e2cf6aa6391b4780b243b8786821ea980d77b01481ba004e8e070f4d08419
prometheus-2.54.0-rc.0.linux-amd64.tar.gz	linux	amd64	100.80 MiB	ca8ff7cdcf7808690779dd15ab065a5bd0d4a745d556a372767fad1cb12337354
prometheus-2.54.0-rc.0.windows-amd64.zip	windows	amd64	103.06 MiB	5d77cc5e2a89d1787c43ae4ce378aba3e03af55383add974946e9376511d38e

- Copy link address and paste it into **prom.sh** file with **sudo wget**

sudo wget

<https://github.com/prometheus/prometheus/releases/download/v2.54.0-rc.0/prometheus-2.54.0-rc.0.linux-amd64.tar.gz>

```
aws Services Search [Alt+S]
sudo wget https://github.com/prometheus/prometheus/releases/download/v2.54.0-rc.0/prometheus-2.54.0-rc.0.linux-amd64.tar.gz
-- INSERT --
```

i-00dc2969aefcc96b (Monitoring)
Public IPs: 3.109.2.121 Private IPs: 172.31.15.199

- Press **esc--->:wq** (For save and exit)
- Now execute below command to run **prom.sh**

\$sh prom.sh

```

← → C  ap-south-1.console.aws.amazon.com/ec2-instance-connect/ssh?region=ap-south-1&connType=standard&instanceId=i-00dcb2969a
aws Services Search [Alt+S]
See "man sudo_root" for details.

ubuntu@ip-172-31-15-199:~$ sudo su
root@ip-172-31-15-199:/home/ubuntu# vi prom.sh
root@ip-172-31-15-199:/home/ubuntu# sh prom.sh
--2024-08-06 17:30:48-- https://github.com/prometheus/prometheus/releases/download/v2.54.0-rc.0/prometheus-
Resolving github.com (github.com) ... 20.207.73.82
Connecting to github.com (github.com) |20.207.73.82|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://objects.githubusercontent.com/github-production-release-asset-2e65be/6838921/100b593f-8b0f
56&X-Amz-Credential=releaseassetproduction%2F20240806%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-Date=20240806T17
3ccb4ae57293116354179d758c81dbfbf5e557c0b3b540cc7&X-Amz-SignedHeaders=host&actor_id=0&key_id=0&repo_id=6838
ame%3Dprometheus-2.54.0-rc.0.linux-amd64.tar.gz&response-content-type=application%2Foctet-stream [following]
--2024-08-06 17:30:49-- https://objects.githubusercontent.com/github-production-release-asset-2e65be/683892
=AWS4-HMAC-SHA256&X-Amz-Credential=releaseassetproduction%2F20240806%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-D
=c90896a3616fd43ccb44ae57293116354179d758c81dbfbf5e557c0b3b540cc7&X-Amz-SignedHeaders=host&actor_id=0&key_id
ment%3B%20filename%3Dprometheus-2.54.0-rc.0.linux-amd64.tar.gz&response-content-type=application%2Foctet-str
Resolving objects.githubusercontent.com (objects.githubusercontent.com) ... 185.199.108.133, 185.199.109.133,
Connecting to objects.githubusercontent.com (objects.githubusercontent.com) |185.199.108.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 105692266 (101M) [application/octet-stream]
Saving to: 'prometheus-2.54.0-rc.0.linux-amd64.tar.gz'

prometheus-2.54.0-rc.0.linux-amd64.tar.g 100%[=====] 2024-08-06 17:30:51 (131 MB/s) - 'prometheus-2.54.0-rc.0.linux-amd64.tar.gz' saved [105692266/105692266]

root@ip-172-31-15-199:/home/ubuntu# []

```

i-00dcb2969aefcc96b (Monitoring)
PublicIPs: 3.109.2.121 PrivateIPs: 172.31.15.199

- Now execute below command for unzip the file

\$ tar -xvzf prometheus-2.54.0-rc.0.linux-amd64.tar.gz

```

aws Services Search [Alt+S]
prometheus-2.54.0-rc.0.linux-amd64.tar.g 100%[=====] 2024-08-06 17:30:51 (131 MB/s) - 'prometheus-2.54.0-rc.0.linux-amd64.tar.gz' saved [105692266/105692266]

root@ip-172-31-15-199:/home/ubuntu# ls
prom.sh prometheus-2.54.0-rc.0.linux-amd64.tar.gz
root@ip-172-31-15-199:/home/ubuntu# tar -xvzf prometheus-2.54.0-rc.0.linux-amd64.tar.gz
prometheus-2.54.0-rc.0.linux-amd64/
prometheus-2.54.0-rc.0.linux-amd64/promtool
prometheus-2.54.0-rc.0.linux-amd64/prometheus.yml
prometheus-2.54.0-rc.0.linux-amd64/consoles/
prometheus-2.54.0-rc.0.linux-amd64/consoles/prometheus-overview.html
prometheus-2.54.0-rc.0.linux-amd64/consoles/node.html
prometheus-2.54.0-rc.0.linux-amd64/consoles/node-disk.html
prometheus-2.54.0-rc.0.linux-amd64/consoles/node-exporter.html.example
prometheus-2.54.0-rc.0.linux-amd64/consoles/node-overview.html
prometheus-2.54.0-rc.0.linux-amd64/consoles/prometheus.html
prometheus-2.54.0-rc.0.linux-amd64/consoles/node-cpu.html
prometheus-2.54.0-rc.0.linux-amd64/NOTICE
prometheus-2.54.0-rc.0.linux-amd64/LICENSE
prometheus-2.54.0-rc.0.linux-amd64/prometheus
prometheus-2.54.0-rc.0.linux-amd64/console_libraries/
prometheus-2.54.0-rc.0.linux-amd64/console_libraries/prom.lib
prometheus-2.54.0-rc.0.linux-amd64/console_libraries/menu.lib
root@ip-172-31-15-199:/home/ubuntu# ls
prometheus-2.54.0-rc.0.linux-amd64  prometheus-2.54.0-rc.0.linux-amd64.tar.gz
root@ip-172-31-15-199:/home/ubuntu# []

i-00dcb2969aefcc96b (Monitoring)
PublicIPs: 3.109.2.121 PrivateIPs: 172.31.15.199

```

- Now go inside to the folder **prometheus-2.54.0-rc.0.linux-amd64** by execute below command

```
$cd prometheus-2.54.0-rc.0.linux-amd64
```

```
root@ip-172-31-15-199:/home/ubuntu# ls
prom.sh  prometheus-2.54.0-rc.0.linux-amd64  prometheus-2.54.0-rc.0.linux-amd64.tar.gz
root@ip-172-31-15-199:/home/ubuntu# cd prometheus-2.54.0-rc.0.linux-amd64
root@ip-172-31-15-199:/home/ubuntu/prometheus-2.54.0-rc.0.linux-amd64# ls
LICENSE  NOTICE  console_libraries  consoles  prometheus  prometheus.yml  promtool
root@ip-172-31-15-199:/home/ubuntu/prometheus-2.54.0-rc.0.linux-amd64#
```

i-00dcb2969aefcc96b (Monitoring)

Public IPs: 3.109.2.121 Private IPs: 172.31.15.199

Step 38: Now download node exporter in Host machine And Master Machine

- For this go to browser and browse official document

https://prometheus.io/download/#node_exporter

File name	OS	Arch	Size	SHA256 Checksum
mysqld_exporter-0.15.1.darwin-amd64.tar.gz	darwin	amd64	7.77 MiB	b270a900be7c3ce2732f719820d54d0d4fb5360725dc0ac651f9682e3f39f7f9
mysqld_exporter-0.15.1.linux-amd64.tar.gz	linux	amd64	7.90 MiB	85ea50c68e1b9f466c1df10ff016652dd210371d42245e012b876265e89ae29d
mysqld_exporter-0.15.1.windows-amd64.zip	windows	amd64	8.18 MiB	556adff778030370c461d059c0b4375e4296472a3af6036d217c979f68bcd5cb4e

node_exporter

Exporter for machine metrics [prometheus/node_exporter](#)

File name	OS	Arch	Size	SHA256 Checksum
node_exporter-1.8.2.darwin-amd64.tar.gz	darwin	amd64	4.83 MiB	97ad998fe48004a0b005f2c5adc6a6eed08309f5fc298a6d3e94ee2e73cf1728
node_exporter-1.8.2.linux-amd64.tar.gz	linux	amd64	10.18 MiB	6800dd0b3ec45fdde992c19071d6b5253aed3ead7bf0636805a51d05c6643c66

promlong

- Now copy the link

[https://github.com/prometheus/node_exporter/releases
/download/v1.8.2/node_exporter-1.8.2.linux-
amd64.tar.gz](https://github.com/prometheus/node_exporter/releases/download/v1.8.2/node_exporter-1.8.2.linux-amd64.tar.gz)

- Now go to Host machine and execute below command
\$wget copied link

\$wget

https://github.com/prometheus/node_exporter/releases/download/v1.8.2/node_exporter-1.8.2.linux-amd64.tar.gz

- Now execute below command to unzip the file

\$ tar xvfz node_exporter-1.8.2.linux-amd64.tar.gz

```
root@ip-172-31-7-75:/home/ubuntu# ls
node_exporter-1.8.2.linux-amd64.tar.gz
root@ip-172-31-7-75:/home/ubuntu# tar xvfz node_exporter-1.8.2.linux-amd64.tar.gz
node_exporter-1.8.2.linux-amd64/
node_exporter-1.8.2.linux-amd64/NOTICE
node_exporter-1.8.2.linux-amd64/node_exporter
node_exporter-1.8.2.linux-amd64/LICENSE
root@ip-172-31-7-75:/home/ubuntu# ls
node_exporter-1.8.2.linux-amd64  node_exporter-1.8.2.linux-amd64.tar.gz
root@ip-172-31-7-75:/home/ubuntu# []
```

i-04c252fe8cff36c7e (Host)

PublicIPs: 35.154.82.10 PrivateIPs: 172.31.7.75

- Now go to the folder

\$cd node_exporter-1.8.2.linux-amd64

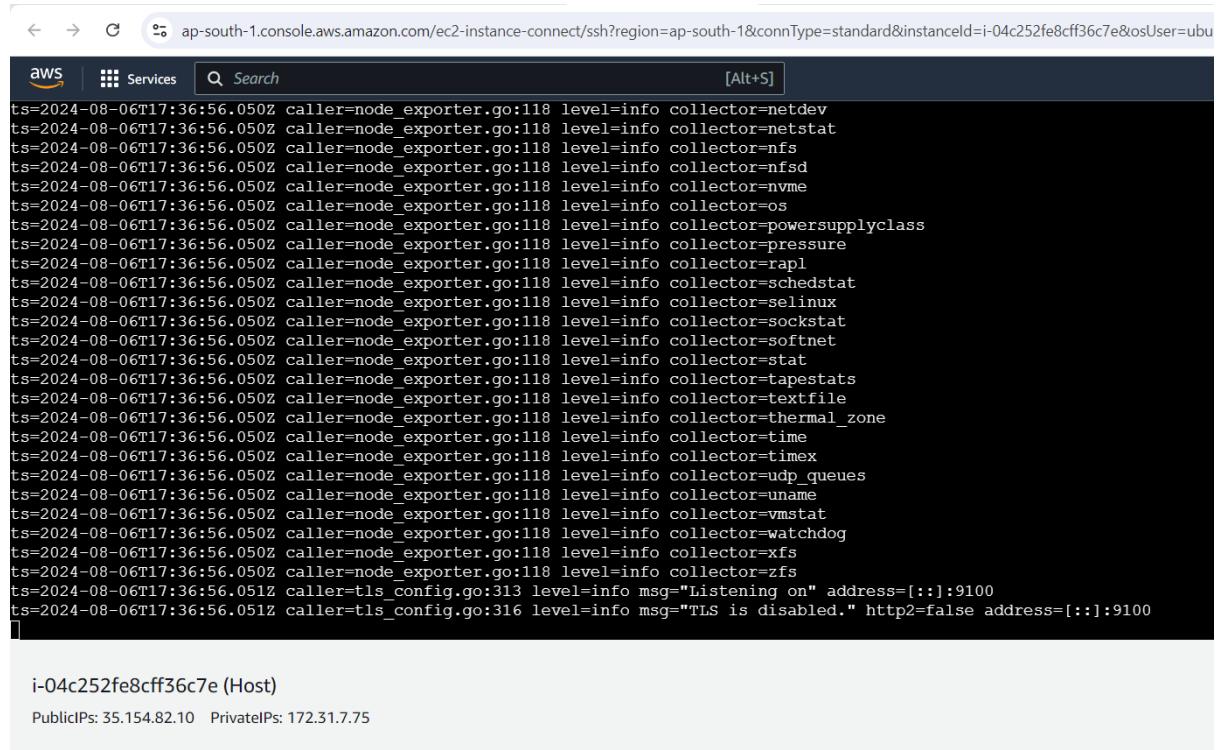
```
root@ip-172-31-7-75:/home/ubuntu# ls
node_exporter-1.8.2.linux-amd64  node_exporter-1.8.2.linux-amd64.tar.gz
root@ip-172-31-7-75:/home/ubuntu# cd node_exporter-1.8.2.linux-amd64
root@ip-172-31-7-75:/home/ubuntu/node_exporter-1.8.2.linux-amd64# ls
LICENSE  NOTICE  node_exporter
root@ip-172-31-7-75:/home/ubuntu/node_exporter-1.8.2.linux-amd64# []
```

i-04c252fe8cff36c7e (Host)

PublicIPs: 35.154.82.10 PrivateIPs: 172.31.7.75

- Now execute below command

\$./node_exporter



The screenshot shows a terminal window titled "aws Services" with a search bar and an "Alt+S" keybinding. The window displays a log of Node Exporter metrics. The log entries are timestamped from August 6, 2024, at 17:36:56.050Z to 17:36:56.051Z. The log shows various collectors being initialized, such as netdev, netstat, nfs, nfssd, nvme, os, powersupplyclass, pressure, rapl, schedstat, selinux, sockstat, softnet, stat, tapestats, textfile, thermal_zone, time, timex, udp_queues, uname, vmstat, watchdog, xfs, and zfs. It also shows a message about listening on port 9100 and TLS being disabled.

```
ts=2024-08-06T17:36:56.050Z caller=node_exporter.go:118 level=info collector=netdev
ts=2024-08-06T17:36:56.050Z caller=node_exporter.go:118 level=info collector=netstat
ts=2024-08-06T17:36:56.050Z caller=node_exporter.go:118 level=info collector=nfs
ts=2024-08-06T17:36:56.050Z caller=node_exporter.go:118 level=info collector=nfssd
ts=2024-08-06T17:36:56.050Z caller=node_exporter.go:118 level=info collector=nvme
ts=2024-08-06T17:36:56.050Z caller=node_exporter.go:118 level=info collector=os
ts=2024-08-06T17:36:56.050Z caller=node_exporter.go:118 level=info collector=powersupplyclass
ts=2024-08-06T17:36:56.050Z caller=node_exporter.go:118 level=info collector=pressure
ts=2024-08-06T17:36:56.050Z caller=node_exporter.go:118 level=info collector=rapl
ts=2024-08-06T17:36:56.050Z caller=node_exporter.go:118 level=info collector=schedstat
ts=2024-08-06T17:36:56.050Z caller=node_exporter.go:118 level=info collector=selinux
ts=2024-08-06T17:36:56.050Z caller=node_exporter.go:118 level=info collector=sockstat
ts=2024-08-06T17:36:56.050Z caller=node_exporter.go:118 level=info collector=softnet
ts=2024-08-06T17:36:56.050Z caller=node_exporter.go:118 level=info collector=stat
ts=2024-08-06T17:36:56.050Z caller=node_exporter.go:118 level=info collector=tapestats
ts=2024-08-06T17:36:56.050Z caller=node_exporter.go:118 level=info collector=textfile
ts=2024-08-06T17:36:56.050Z caller=node_exporter.go:118 level=info collector=thermal_zone
ts=2024-08-06T17:36:56.050Z caller=node_exporter.go:118 level=info collector=time
ts=2024-08-06T17:36:56.050Z caller=node_exporter.go:118 level=info collector=timex
ts=2024-08-06T17:36:56.050Z caller=node_exporter.go:118 level=info collector=udp_queues
ts=2024-08-06T17:36:56.050Z caller=node_exporter.go:118 level=info collector=uname
ts=2024-08-06T17:36:56.050Z caller=node_exporter.go:118 level=info collector=vmstat
ts=2024-08-06T17:36:56.050Z caller=node_exporter.go:118 level=info collector=watchdog
ts=2024-08-06T17:36:56.050Z caller=node_exporter.go:118 level=info collector=xfs
ts=2024-08-06T17:36:56.051Z caller=tls_config.go:313 level=info msg="Listening on" address=[::]:9100
ts=2024-08-06T17:36:56.051Z caller=tls_config.go:316 level=info msg="TLS is disabled." http2=false address=[::]:9100
[]
```

i-04c252fe8cff36c7e (Host)
PublicIPs: 35.154.82.10 PrivateIPs: 172.31.7.75

- **For verify go to browser and browse**

Publicip of Host machine instance:9100



- **Public ip of instance:9100/metrics**

← → C △ Not secure 35.154.82.10:9100/metrics

```
# HELP go_gc_duration_seconds A summary of the pause duration of garbage collection cycles.
# TYPE go_gc_duration_seconds summary
go_gc_duration_seconds{quantile="0"} 0
go_gc_duration_seconds{quantile="0.25"} 0
go_gc_duration_seconds{quantile="0.5"} 0
go_gc_duration_seconds{quantile="0.75"} 0
go_gc_duration_seconds{quantile="1"} 0
go_gc_duration_seconds_sum 0
go_gc_duration_seconds_count 0
# HELP go_goroutines Number of goroutines that currently exist.
# TYPE go_goroutines gauge
go_goroutines 7
# HELP go_info Information about the Go environment.
# TYPE go_info gauge
go_info{version="go1.22.5"} 1
# HELP go_memstats_alloc_bytes Number of bytes allocated and still in use.
# TYPE go_memstats_alloc_bytes gauge
go_memstats_alloc_bytes 834440
# HELP go_memstats_alloc_bytes_total Total number of bytes allocated, even if freed.
# TYPE go_memstats_alloc_bytes_total counter
go_memstats_alloc_bytes_total 834440
# HELP go_memstats_buck_hash_sys_bytes Number of bytes used by the profiling bucket hash table.
# TYPE go_memstats_buck_hash_sys_bytes gauge
go_memstats_buck_hash_sys_bytes 1.448924e+06
# HELP go_memstats_frees_total Total number of frees.
# TYPE go_memstats_frees_total counter
go_memstats_frees_total 730
# HELP go_memstats_gc_sys_bytes Number of bytes used for garbage collection system metadata.
# TYPE go_memstats_gc_sys_bytes gauge
go_memstats_gc_sys_bytes 1.80124e+06
# HELP go_memstats_heap_alloc_bytes Number of heap bytes allocated and still in use.
# TYPE go_memstats_heap_alloc_bytes gauge
go_memstats_heap_alloc_bytes 834440
# HELP go_memstats_heap_idle_bytes Number of heap bytes waiting to be used.
# TYPE go_memstats_heap_idle_bytes gauge
go_memstats_heap_idle_bytes 1.589248e+06
# HELP go_memstats_heap_inuse_bytes Number of heap bytes that are in use.
# TYPE go_memstats_heap_inuse_bytes gauge
go_memstats_heap_inuse_bytes 2.277376e+06
# HELP go_memstats_heap_objects Number of allocated objects.
# TYPE go_memstats_heap_objects gauge
go_memstats_heap_objects 8112
# HELP go_memstats_heap_released_bytes Number of heap bytes released to OS.
# TYPE go_memstats_heap_released_bytes gauge
go_memstats_heap_released_bytes 1.589248e+06
```

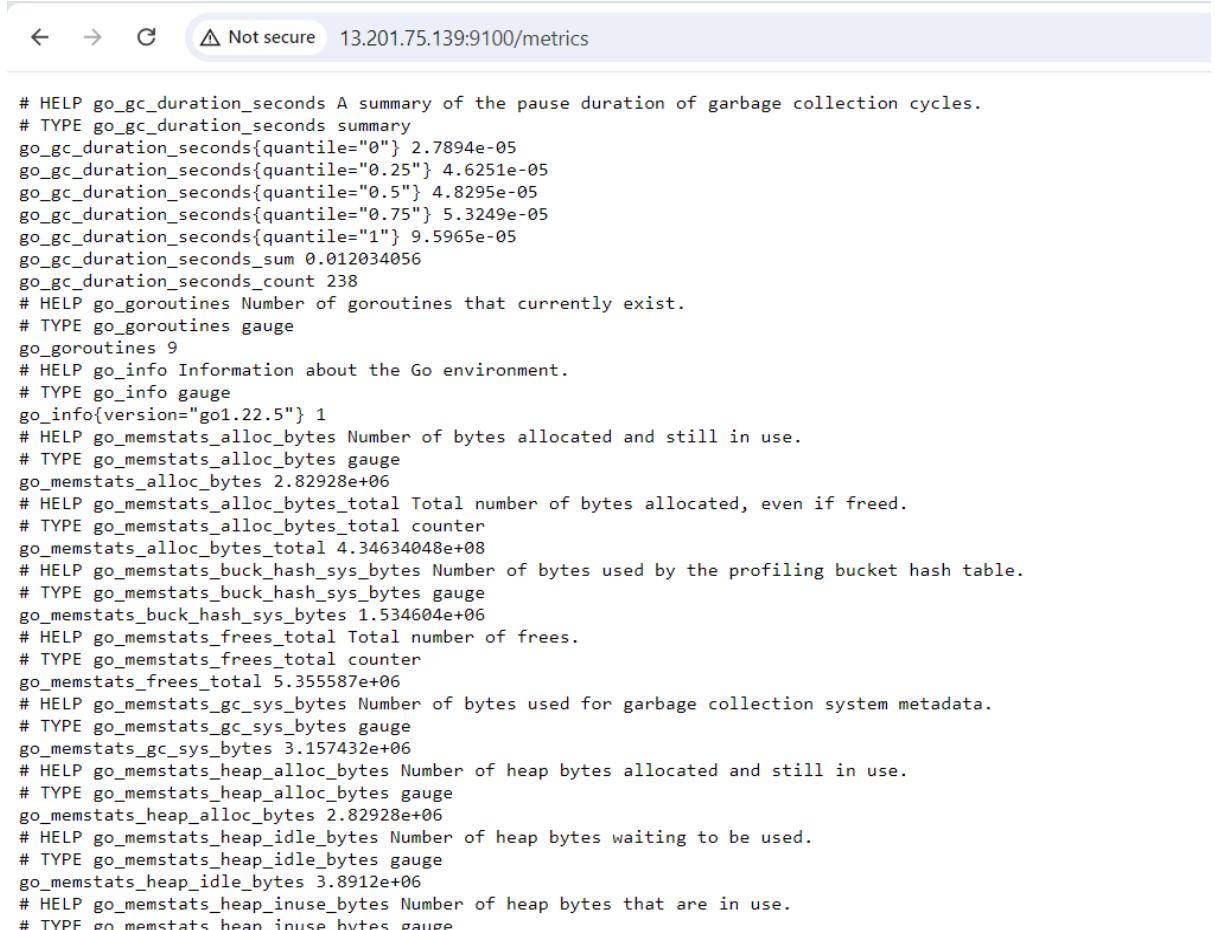
- **It shows all the metrics**
- **You can see we have configured Node Exporter in Host Machine-**

```
root@ip-172-31-7-75:/home/ubuntu# ls
node_exporter-1.8.2.linux-amd64  node_exporter-1.8.2.linux-amd64.tar.gz
root@ip-172-31-7-75:/home/ubuntu# []
```

i-04c252fe8cff36c7e (Host)

- By using same steps we installed Node Exporter in master machine also

Publicip of Master machine instance:9100



```
# HELP go_gc_duration_seconds A summary of the pause duration of garbage collection cycles.
# TYPE go_gc_duration_seconds summary
go_gc_duration_seconds{quantile="0"} 2.7894e-05
go_gc_duration_seconds{quantile="0.25"} 4.6251e-05
go_gc_duration_seconds{quantile="0.5"} 4.8295e-05
go_gc_duration_seconds{quantile="0.75"} 5.3249e-05
go_gc_duration_seconds{quantile="1"} 9.5965e-05
go_gc_duration_seconds_sum 0.012034056
go_gc_duration_seconds_count 238
# HELP go_goroutines Number of goroutines that currently exist.
# TYPE go_goroutines gauge
go_goroutines 9
# HELP go_info Information about the Go environment.
# TYPE go_info gauge
go_info{version="go1.22.5"} 1
# HELP go_memstats_alloc_bytes Number of bytes allocated and still in use.
# TYPE go_memstats_alloc_bytes gauge
go_memstats_alloc_bytes 2.82928e+06
# HELP go_memstats_alloc_bytes_total Total number of bytes allocated, even if freed.
# TYPE go_memstats_alloc_bytes_total counter
go_memstats_alloc_bytes_total 4.34634048e+08
# HELP go_memstats_buck_hash_sys_bytes Number of bytes used by the profiling bucket hash table.
# TYPE go_memstats_buck_hash_sys_bytes gauge
go_memstats_buck_hash_sys_bytes 1.534604e+06
# HELP go_memstats_frees_total Total number of frees.
# TYPE go_memstats_frees_total counter
go_memstats_frees_total 5.355587e+06
# HELP go_memstats_gc_sys_bytes Number of bytes used for garbage collection system metadata.
# TYPE go_memstats_gc_sys_bytes gauge
go_memstats_gc_sys_bytes 3.157432e+06
# HELP go_memstats_heap_alloc_bytes Number of heap bytes allocated and still in use.
# TYPE go_memstats_heap_alloc_bytes gauge
go_memstats_heap_alloc_bytes 2.82928e+06
# HELP go_memstats_heap_idle_bytes Number of heap bytes waiting to be used.
# TYPE go_memstats_heap_idle_bytes gauge
go_memstats_heap_idle_bytes 3.8912e+06
# HELP go_memstats_heap_inuse_bytes Number of heap bytes that are in use.
# TYPE go_memstats_heap_inuse_bytes gauge
```

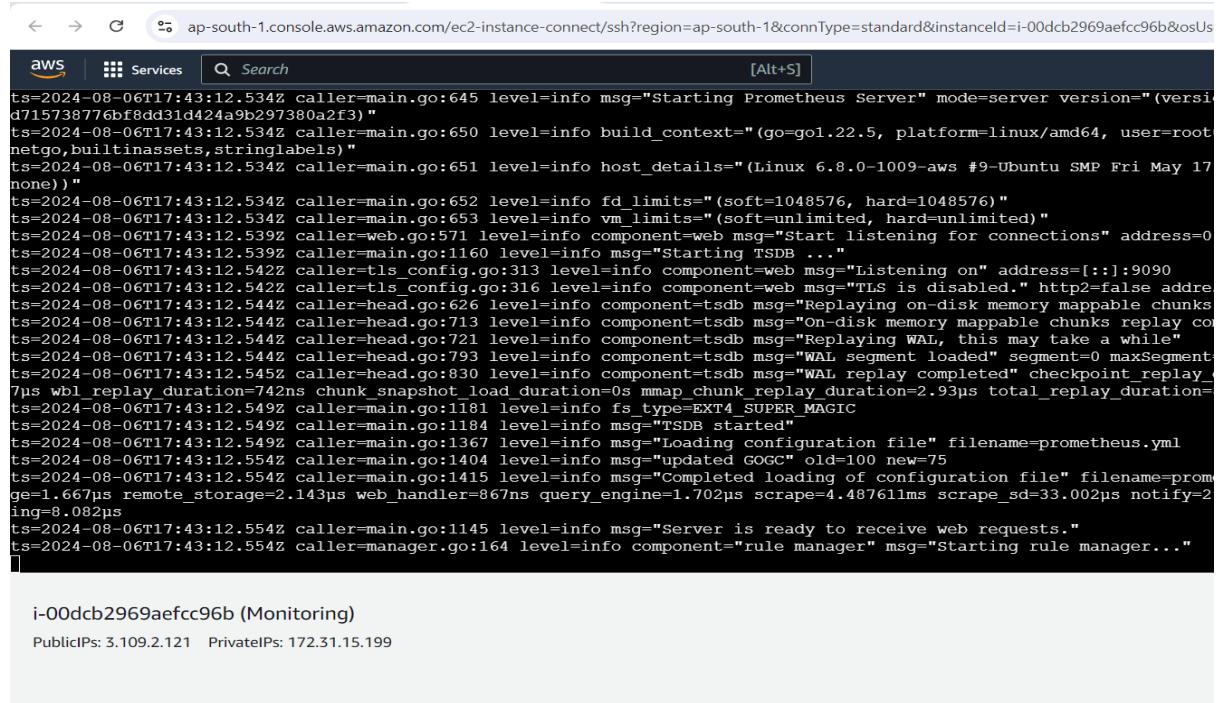
- You can see we have Installed Node Exporter in Master Machine

```
ubuntu@ip-172-31-5-152:~$ ls
ansible.sh jenkins.sh node_exporter-1.8.2.linux-amd64 node_exporter-1.8.2.linux-amd64.tar.gz
ubuntu@ip-172-31-5-152:~$ ]
```

i-0f5f05c6c2c216ee9 (Master)

Step 39: Now go back to the monitoring server and execute below command

\$./ Prometheus



The screenshot shows a terminal window titled 'ap-south-1.console.aws.amazon.com/ec2-instance-connect/ssh?region=ap-south-1&connType=standard&instanceId=i-00dcb2969aefcc96b&osUs'. The terminal displays a continuous stream of log messages from the Prometheus server. The logs include details about the server starting up, listening on port 9090, and performing WAL replay operations. Key log entries include:

```
ts=2024-08-06T17:43:12.534Z caller=main.go:645 level=info msg="Starting Prometheus Server" mode=server version="(version d715738776bf8dd31d424a9b297380a2f3)"  
ts=2024-08-06T17:43:12.534Z caller=main.go:650 level=info build_context="(go=go1.22.5, platform=linux/amd64, user=root netgo,builtinassets,stringlabels)"  
ts=2024-08-06T17:43:12.534Z caller=main.go:651 level=info host_details="(Linux 6.8.0-1009-aws #9-Ubuntu SMP Fri May 17 none)"  
ts=2024-08-06T17:43:12.534Z caller=main.go:652 level=info fd_limits="(soft=1048576, hard=1048576)"  
ts=2024-08-06T17:43:12.534Z caller=main.go:653 level=info vm_limits="(soft=unlimited, hard=unlimited)"  
ts=2024-08-06T17:43:12.539Z caller=web.go:571 level=info component=web msg="Start listening for connections" address=0  
ts=2024-08-06T17:43:12.539Z caller=main.go:1160 level=info msg="Starting TSDB ..."  
ts=2024-08-06T17:43:12.542Z caller=tsl_config.go:313 level=info component=web msg="Listening on" address=[::]:9090  
ts=2024-08-06T17:43:12.542Z caller=tls_config.go:316 level=info component=web msg="TLS is disabled." http2=false address=0  
ts=2024-08-06T17:43:12.544Z caller=head.go:626 level=info component=tsdb msg="Replaying on-disk memory mappable chunks"  
ts=2024-08-06T17:43:12.544Z caller=head.go:713 level=info component=tsdb msg="On-disk memory mappable chunks replay complete"  
ts=2024-08-06T17:43:12.544Z caller=head.go:721 level=info component=tsdb msg="Replaying WAL, this may take a while"  
ts=2024-08-06T17:43:12.544Z caller=head.go:793 level=info component=tsdb msg="WAL segment loaded" segment=0 maxSegment=1  
ts=2024-08-06T17:43:12.545Z caller=head.go:830 level=info component=tsdb msg="WAL replay completed" checkpoint_replay_7us_wbl_replay_duration=742ns chunk_snapshot_load_duration=0s mmap_chunk_replay_duration=2.93us total_replay_duration=0s  
ts=2024-08-06T17:43:12.549Z caller=main.go:1181 level=info fs_type=EXT4_SUPER_MAGIC  
ts=2024-08-06T17:43:12.549Z caller=main.go:1184 level=info msg="TSDB started"  
ts=2024-08-06T17:43:12.549Z caller=main.go:1367 level=info msg="Loading configuration file" filename=prometheus.yml  
ts=2024-08-06T17:43:12.554Z caller=main.go:1404 level=info msg="updated GOGC old=100 new=75  
ts=2024-08-06T17:43:12.554Z caller=main.go:1415 level=info msg="Completed loading of configuration file" filename=prometheus.yml  
remote_storage=2.143us web_handler=867ns query_engine=1.702us scrape=4.487611ms scrape_sd=33.002us notify=2  
ing=8.082us  
ts=2024-08-06T17:43:12.554Z caller=main.go:1145 level=info msg="Server is ready to receive web requests."  
ts=2024-08-06T17:43:12.554Z caller=manager.go:164 level=info component="rule manager" msg="Starting rule manager..."  
[]
```

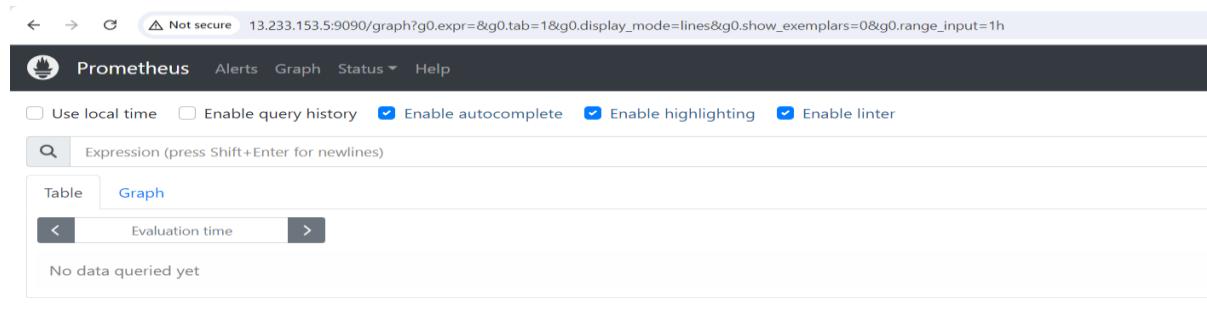
i-00dcb2969aefcc96b (Monitoring)
Public IPs: 3.109.2.121 Private IPs: 172.31.15.199

Step 40: Now copy the same public ip of monitoring instance with Prometheus default port 9090, search on browser

Public ip:9090

----> it shows Prometheus

13.233.153.5:9090



- Now click on status then targets

Prometheus Alerts Graph Status▼ Help

Use local time Enable query history

Q Expression (press Shift+Enter for new line)

Table Graph

< Evaluation time >

No data queried yet

Runtime & Build Information
TSDB Status
Command-Line Flags
Configuration
Rules
Targets
Service Discovery

Enable highlighting Enable linter

Add Panel

- We can see the state is up

Prometheus Alerts Graph Status▼ Help

Targets

All scrape pools ▾ All Unhealthy Collapse All Filter by endpoint or labels

> **prometheus (1/1 up)** show less

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
http://localhost:9090/metrics	UP	instance="localhost:9090" job="prometheus" ↗	10.555s ago	5.505ms	none

- Now go to the prometheus.yml file be execute below command

\$vi Prometheus.yml

- Now define the target in this file, for this we edited the file

- Add below code at last,

-- job_name: "Master"

```
# metrics_path defaults to '/metrics'
```

```
# scheme defaults to 'http'.
```

static_configs:

- targets: ["13.201.75.139:9100"]

- job_name: "Host"

```
# metrics_path defaults to '/metrics'
```

```
# scheme defaults to 'http'.
```

static_configs:

- targets: ["65.0.27.52:9100"]

- Above ip is Host machine ip in which node exporter installed and 9100 is node selector port

The screenshot shows a terminal window within the AWS Cloud9 IDE. The terminal displays a YAML configuration file for Prometheus. The configuration defines four scrape configurations:

- A job named "prometheus" with targets at "localhost:9090".
- A job named "Master" with targets at "13.201.75.139:9100".
- A job named "Host" with targets at "65.0.27.52:9100".

The configuration uses static targets and includes comments explaining the defaults for metrics path and scheme.

```
# A scrape configuration containing exactly one endpoint to scrape:  
# Here it's Prometheus itself.  
scrape_configs:  
  # The job name is added as a label `job=<job_name>` to any timeseries scraped from this config.  
  - job_name: "prometheus"  
  
    # metrics_path defaults to '/metrics'  
    # scheme defaults to 'http'.  
  
    static_configs:  
      - targets: ["localhost:9090"]  
  
  - job_name: "Master"  
  
    # metrics_path defaults to '/metrics'  
    # scheme defaults to 'http'.  
  
    static_configs:  
      - targets: ["13.201.75.139:9100"]  
  
  - job_name: "Host"  
  
    # metrics_path defaults to '/metrics'  
    # scheme defaults to 'http'.  
  
    static_configs:  
      - targets: ["65.0.27.52:9100"]  
"prometheus.yml" 45L, 1251B
```

i-00dcb2969aefcc96b (Monitoring)

PublicIPs: 13.233.153.5 PrivateIPs: 172.31.15.199

- **Save this file (press esc -----> :wq)**

Step 41: Now execute below command to run Prometheus

\$./Prometheus

The screenshot shows a terminal window within the AWS CloudWatch interface. The URL in the address bar is `ap-south-1.console.aws.amazon.com/ec2-instance-connect/ssh?region=ap-south-1&connType=standard&instanceId=i-00dcb2969aefcc96b&osUser=ubuntu&sshPort=22#`. The terminal window has a dark theme with white text. It displays the log output of the `./Prometheus` command. The log includes various system and application logs, such as build context, host details, and Prometheus component startup messages. At the bottom of the terminal, there is a message indicating the server is ready to receive web requests.

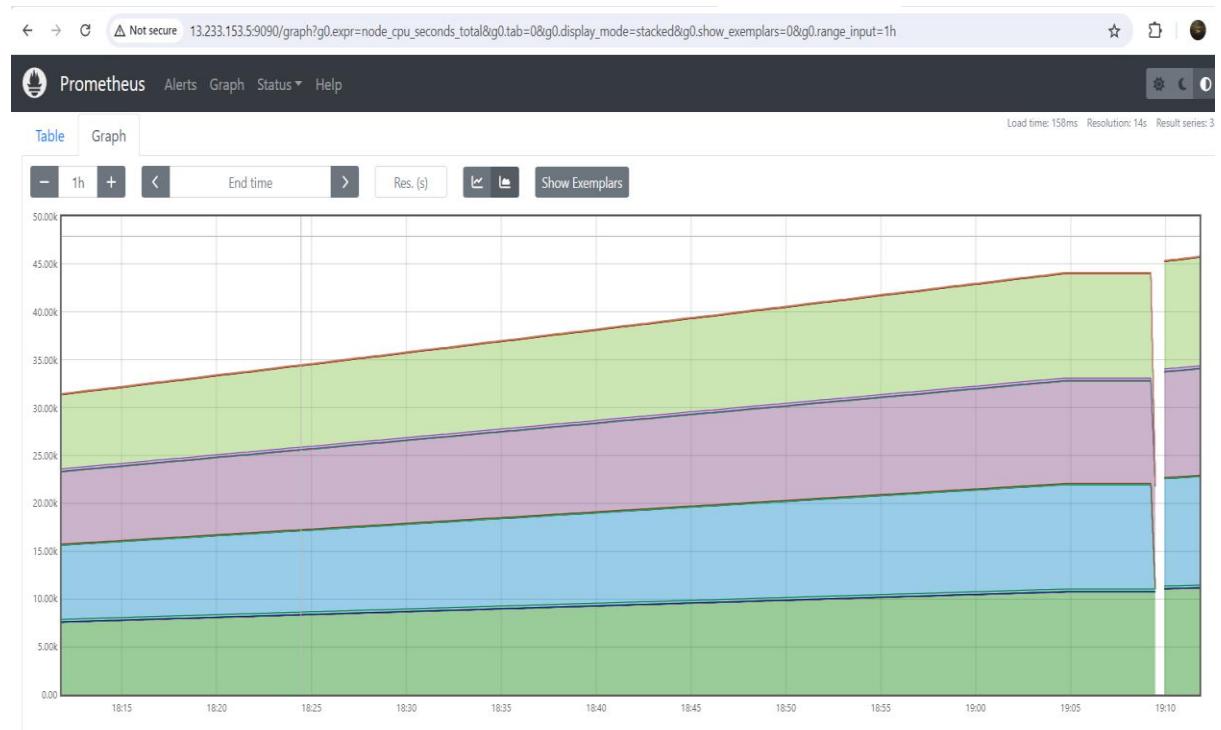
```
d715738776bf8dd31d424a9b297380a2f3)"  
ts=2024-08-07T19:09:34.98Z caller=main.go:650 level=info build_context="(go=go1.22.5, platform=linux/amd64, user=root@55e355ec479f, date=20240730-09:46:56, netgo,builtinassets,stringlabels)"  
ts=2024-08-07T19:09:34.982Z caller=main.go:651 level=info host_details="(Linux 6.8.0-1009-aws #9-Ubuntu SMP Fri May 17 14:39:23 UTC 2024 x86_64 ip-172-31-15-1 none)"  
ts=2024-08-07T19:09:34.982Z caller=main.go:652 level=info fd_limits="(soft=1048576, hard=1048576)"  
ts=2024-08-07T19:09:34.982Z caller=main.go:653 level=info vm_limits="(soft=unlimited, hard=unlimited)"  
ts=2024-08-07T19:09:34.988Z caller=web.go:571 level=info component=web msg="Start listening for connections" address=0.0.0.0:9090  
ts=2024-08-07T19:09:34.988Z caller=main.go:1160 level=info msg="Starting TSDB ..."  
ts=2024-08-07T19:09:34.991Z caller=tls_config.go:313 level=info component=web msg="Listening on" address=[::]:9090  
ts=2024-08-07T19:09:34.991Z caller=tls_config.go:316 level=info component=web msg="TLS is disabled." http2=false address=[::]:9090  
ts=2024-08-07T19:09:34.993Z caller=head.go:626 level=info component=tsdb msg="Replaying on-disk memory mappable chunks if any"  
ts=2024-08-07T19:09:34.995Z caller=head.go:713 level=info component=tsdb msg="On-disk memory mappable chunks replay completed" duration=1.75878ms  
ts=2024-08-07T19:09:34.995Z caller=head.go:721 level=info component=tsdb msg="Replaying WAL, this may take a while"  
ts=2024-08-07T19:09:35.029Z caller=head.go:793 level=info component=tsdb msg="WAL segment loaded" segment=0 maxSegment=1  
ts=2024-08-07T19:09:35.029Z caller=head.go:793 level=info component=tsdb msg="WAL segment loaded" segment=1 maxSegment=1  
ts=2024-08-07T19:09:35.029Z caller=head.go:830 level=info component=tsdb msg="WAL replay completed" checkpoint_replay_duration=44.235µs wal_replay_duration=34.885ms wbl_replay_duration=3.006µs chunk_snapshot_load_duration=0s mmap_chunk_replay_duration=1.75878ms total_replay_duration=36.439512ms  
ts=2024-08-07T19:09:35.032Z caller=main.go:1181 level=info fs_type=EXT4_SUPER_MAGIC  
ts=2024-08-07T19:09:35.032Z caller=main.go:1184 level=info msg="TSDB started"  
ts=2024-08-07T19:09:35.032Z caller=main.go:1367 level=info msg="Loading configuration file" filename=prometheus.yml  
ts=2024-08-07T19:09:35.037Z caller=main.go:1404 level=info msg="updated GOGC old=100 new=75  
ts=2024-08-07T19:09:35.038Z caller=main.go:1415 level=info msg="Completed loading of configuration file" filename=prometheus.yml totalDuration=5.330539ms db_size=2.113µs remote_storage=2.57µs web_handler=1.238µs query_engine=1.739µs scrape=4.228472ms scrape_sd=65.543µs notify=51.164µs notify_sd=29.153µs rules=3.041 racing=74.787µs  
ts=2024-08-07T19:09:35.038Z caller=main.go:1145 level=info msg="Server is ready to receive web requests."  
ts=2024-08-07T19:09:35.038Z caller=manager.go:164 level=info component="rule manager" msg="Starting rule manager..."  
[]  
  
i-00dcb2969aefcc96b (Monitoring)  
PublicIPs: 13.233.153.5 PrivateIPs: 172.31.15.199
```

Step 42: Now go to the Prometheus page on browser and refresh the page

The screenshot shows the Prometheus Targets page with three sections: Host (1/1 up), Master (1/1 up), and prometheus (1/1 up). Each section displays a table with columns: Endpoint, State, Labels, Last Scrape, Scrape Duration, and Error. All targets are marked as UP and have been scraped successfully.

Host (1/1 up)	Master (1/1 up)	prometheus (1/1 up)
http://65.0.27.52:9100/metrics	http://13.201.75.139:9100/metrics	http://localhost:9090/metrics
UP	UP	UP
instance="65.0.27.52:9100" job="Host"	instance="13.201.75.139:9100" job="Master"	instance="localhost:9090" job="prometheus"
6.141s ago	10.121s ago	15.3s ago
15.096ms	14.462ms	5.834ms

- We can also see the graph for this click on graph
- Now type node_cpu_seconds_total (for cpu utilization), then click on graph



Step 43: Now installing graphana for create dashboard

- For install Grafana go to browser and search install grafana for ubuntu
- Go to official website and scroll down ,you will get the command s for install Grafana

The screenshot shows a web browser window with several tabs open at the top:

- aws_instance | Resources | hashicorp.com
- Build log [#4] [Jenkins]
- 65.273.42:9100/metrics
- Prometheus Time Series Collector
- Install Grafana on Debian or Ubuntu

The main content area displays the Grafana documentation for "latest/setup-grafana/installation/debian". On the left, a sidebar menu is expanded under "Set up" to show "Install Grafana" options for "Debian or Ubuntu", "RHEL or Fedora", "SUSE or openSUSE", "Grafana Docker image", "Grafana on Kubernetes", "Grafana on Helm Charts", "macOS", and "Windows".

The main content area includes a table of repository options:

Grafana Enterprise (Beta)	grafana-enterprise	https://apt.grafana.com beta main
Grafana OSS	grafana	https://apt.grafana.com stable main
Grafana OSS (Beta)	grafana	https://apt.grafana.com beta main

A "NOTE" section states: "Grafana Enterprise is the recommended and default edition. It is available for free and includes all the features of the OSS edition. You can also upgrade to the [full Enterprise feature set](#), which has support for Enterprise plugins."

Below the note, instructions say: "Complete the following steps to install Grafana from the APT repository:"

- 1 Install the prerequisite packages:

```
bash
sudo apt-get install -y apt-transport-https software-properties-common wget
```
- 2 Import the GPG key:

```
bash
sudo mkdir -p /etc/apt/keyrings/
wget -q -O - https://apt.grafana.com/gpg.key | gpg --dearmor | sudo tee /etc/apt/keyrings/grafana.gpg
```
- 3 To add a repository for stable releases, run the following command:

- We will install Grafana in the same machine in which we installed Prometheus

- Go to the Host machine and come back one step back by execute (`$cd ..`) and then create a file by execute below command

\$vi grafana.sh

Now copy and paste one by one command in grafana.sh file

- Now Execute below command to run this file

\$sh grafana.sh

```
aws | Services | Search [Alt+S]
Processing triggers for man-db (2.12.0-4build2) ...
Processing triggers for sgml-base (1.31) ...
Setting up libfontconfig1:amd64 (2.15.0-1ubuntu2) ...
Setting up grafana (11.1.3) ...
Info: Selecting UID from range 100 to 999 ...

Info: Adding system user 'grafana' (UID 111) ...
Info: Adding new user 'grafana' (UID 111) with group 'grafana' ...
Info: Not creating home directory '/usr/share/grafana'.
## NOT starting on installation, please execute the following statements to configure
sudo /bin/systemctl daemon-reload
sudo /bin/systemctl enable grafana-server
## You can start grafana-server by executing
sudo /bin/systemctl start grafana-server
Processing triggers for libdc-bin (2.39-0ubuntu0.2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
```

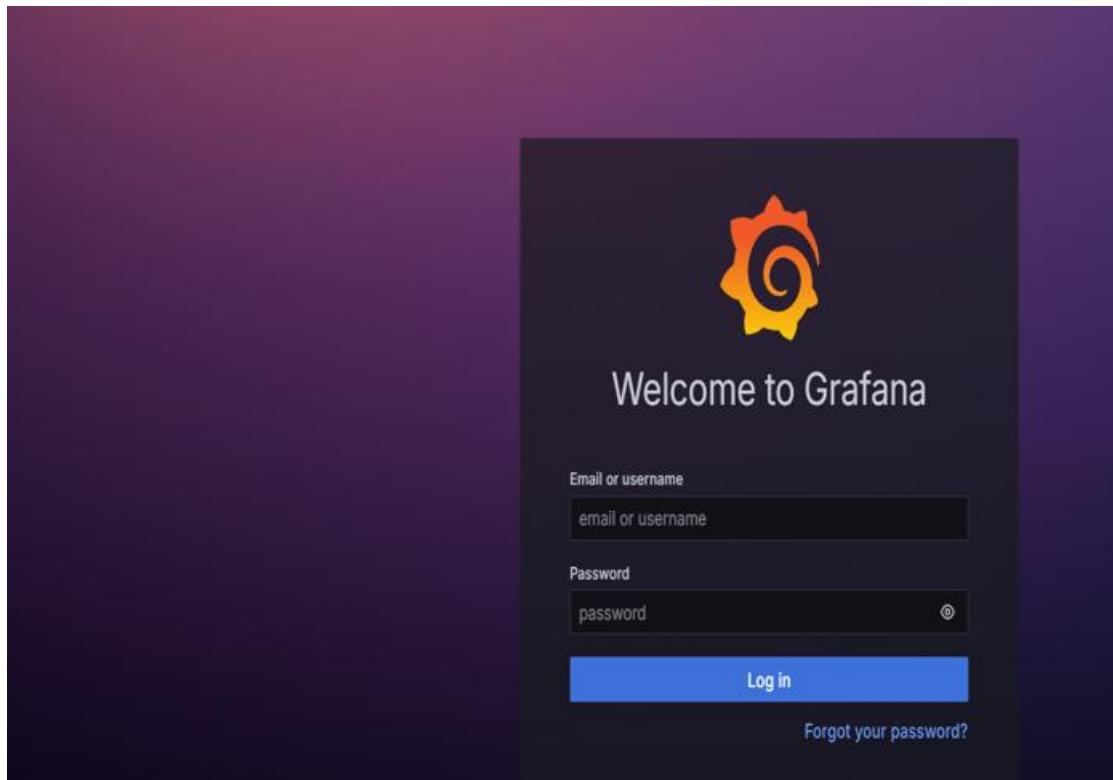
Step 43: Now execute below command in same instance

\$sudo /bin/systemctl start grafana-server

```
No VM guests are running outdated hypervisor (qemu) binaries on this host.  
root@ip-172-31-15-199:/home/ubuntu# sudo /bin/systemctl start grafana-server  
root@ip-172-31-15-199:/home/ubuntu#
```

- Now copy the public ip of Monitoring machine instance and browse it

13.233.153.5:3000 (Grafana default port is 3000)

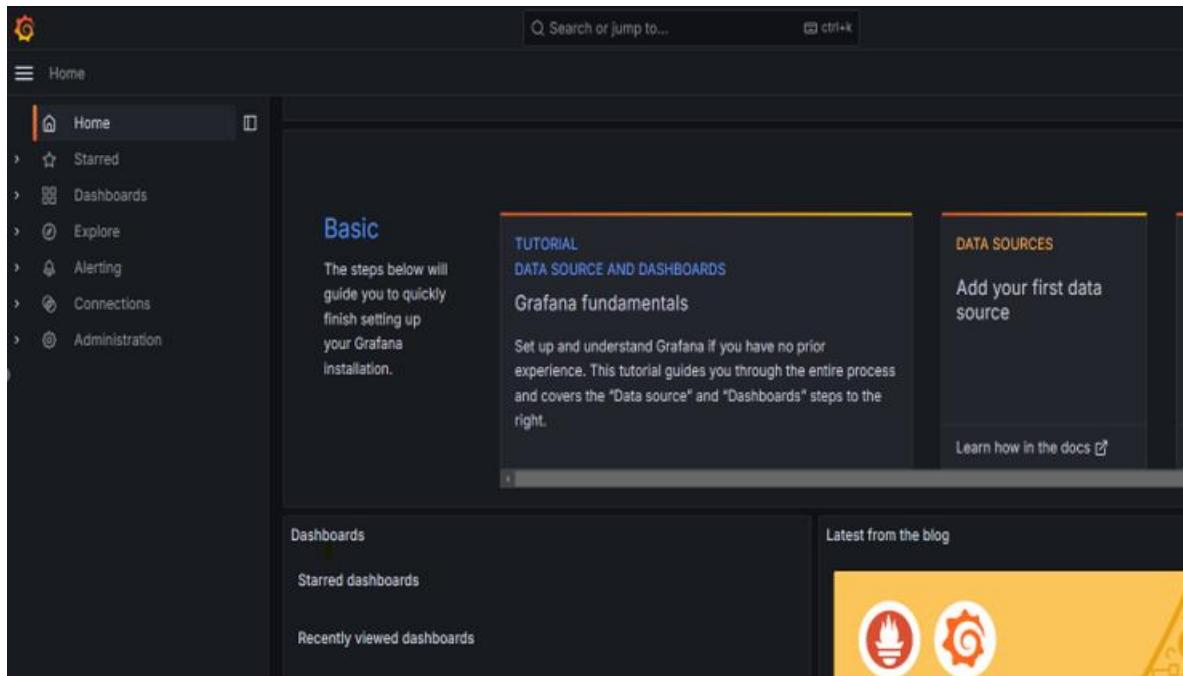


- You can see the Grafana login page
- For login

Use user name: admin

Use password : admin

- Then click on skip



- You can see the grafana home page

Step 44: Now again in Monitoring Machine execute below command to run Prometheus

\$ls

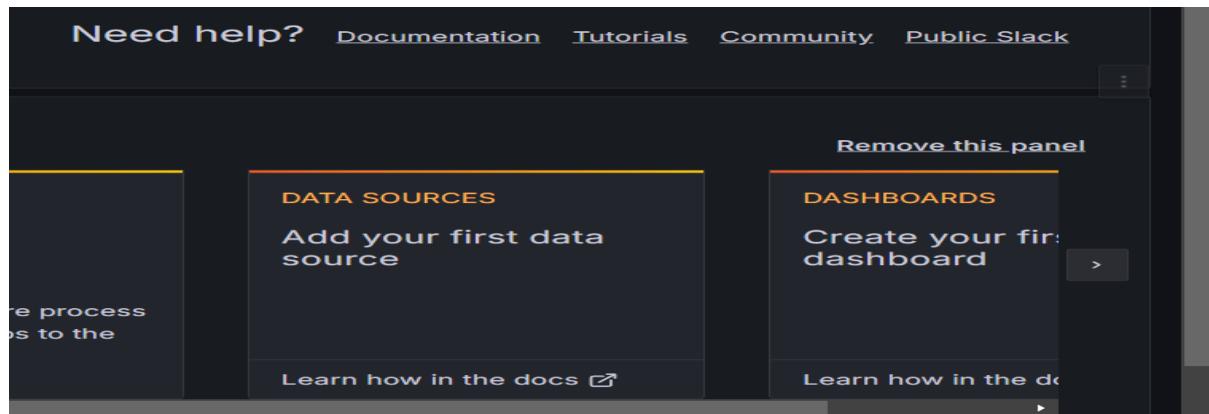
\$cd Prometheus-2.54.0-rc.0.linux-amd64

\$.Prometheus

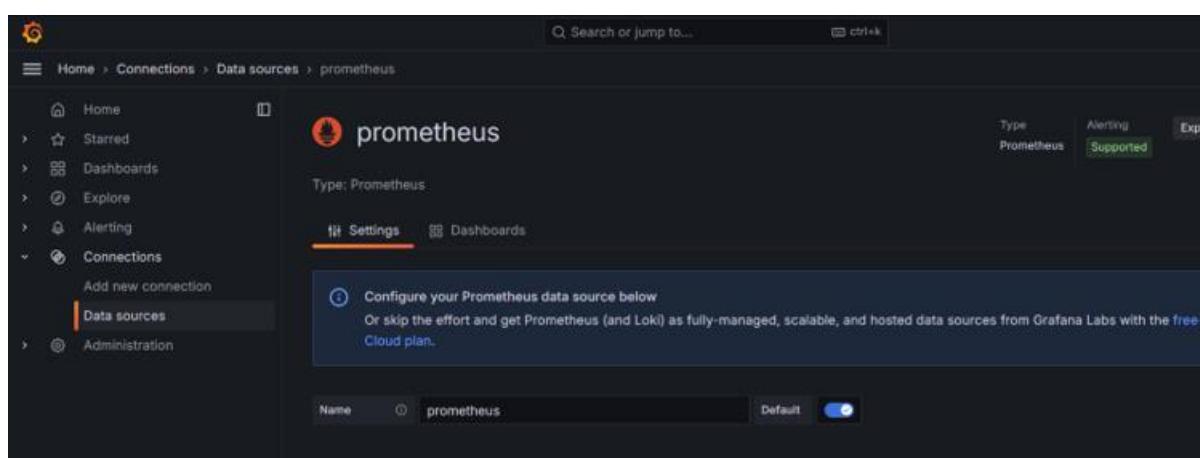
```
← → C ap-south-1.console.aws.amazon.com/ec2-instance-connect/ssh?region=ap-south-1&connType=standard&instanceId=i-00dc2969aefcc96b&osUser=u
aws | Services | Search [Alt+S]
ts=2024-08-07T19:22:25.414Z caller=main.go:650 level=info build_context="(go=go1.22.5, platform=linux/amd64, user=root@55.117.11.11)" netgo.builtinassets.stringlabels)
ts=2024-08-07T19:22:25.414Z caller=main.go:651 level=info host_details="(Linux 6.8.0-1009-aws #9-Ubuntu SMP Fri May 17 14:00:00 UTC 2024 root@i-00dc2969aefcc96b ~)"
ts=2024-08-07T19:22:25.414Z caller=main.go:653 level=info vm_limits="(soft=unlimited, hard=unlimited)"
ts=2024-08-07T19:22:25.419Z caller=web.go:571 level=info component=web msg="Start listening for connections" address=0.0.0.0:9090
ts=2024-08-07T19:22:25.419Z caller=main.go:1160 level=info msg="Starting TSDB ..."
ts=2024-08-07T19:22:25.421Z caller=tls_config.go:313 level=info component=web msg="Listening on" address=[::]:9090
ts=2024-08-07T19:22:25.421Z caller=tls_config.go:316 level=info component=web msg="TLS is disabled." http2=false address=0.0.0.0:9090
ts=2024-08-07T19:22:25.424Z caller=head.go:626 level=info component=tsdb msg="Replaying on-disk memory mappable chunks if any"
ts=2024-08-07T19:22:25.428Z caller=head.go:713 level=info component=tsdb msg="On-disk memory mappable chunks replay completed"
ts=2024-08-07T19:22:25.428Z caller=head.go:721 level=info component=tsdb msg="Replaying WAL, this may take a while"
ts=2024-08-07T19:22:25.460Z caller=head.go:793 level=info component=tsdb msg="WAL segment loaded" segment=0 maxSegment=2
ts=2024-08-07T19:22:25.465Z caller=head.go:793 level=info component=tsdb msg="WAL segment loaded" segment=1 maxSegment=2
ts=2024-08-07T19:22:25.466Z caller=head.go:793 level=info component=tsdb msg="WAL segment loaded" segment=2 maxSegment=2
ts=2024-08-07T19:22:25.466Z caller=head.go:830 level=info component=tsdb msg="WAL replay completed" checkpoint_replay_duration=0.037ms wbl_replay_duration=750ns chunk_snapshot_load_duration=0s mmap_chunk_replay_duration=3.157301ms total_replay_duration=0.037ms
ts=2024-08-07T19:22:25.472Z caller=main.go:1181 level=info fs_type=EXT4_SUPER_MAGIC
ts=2024-08-07T19:22:25.472Z caller=main.go:1184 level=info msg="TSDB started"
ts=2024-08-07T19:22:25.472Z caller=main.go:1367 level=info msg="Loading configuration file" filename=prometheus.yml
ts=2024-08-07T19:22:25.477Z caller=main.go:1404 level=info msg="updated GOGC" old=100 new=75
ts=2024-08-07T19:22:25.477Z caller=main.go:1415 level=info msg="Completed loading of configuration file" filename=prometheus.yml
remote_storage=2.298μs web_handler=1.092μs query_engine=1.458μs scrape=3.978946ms scrape_sd=78.342μs notify=36acing=7.191μs
ts=2024-08-07T19:22:25.477Z caller=main.go:1145 level=info msg="Server is ready to receive web requests."
ts=2024-08-07T19:22:25.477Z caller=manager.go:164 level=info component="rule manager" msg="Starting rule manager..."
```

i-00dc2969aefcc96b (Monitoring)
PublicIPs: 13.233.153.5 PrivateIPs: 172.31.15.199

Step 45: Now click on Data source(add your first data source)



- Then select the Prometheus as data source



- Name as Prometheus-sumit
- Now copy the link of Prometheus and paste it into connection section

13.233.153.5 :9090/

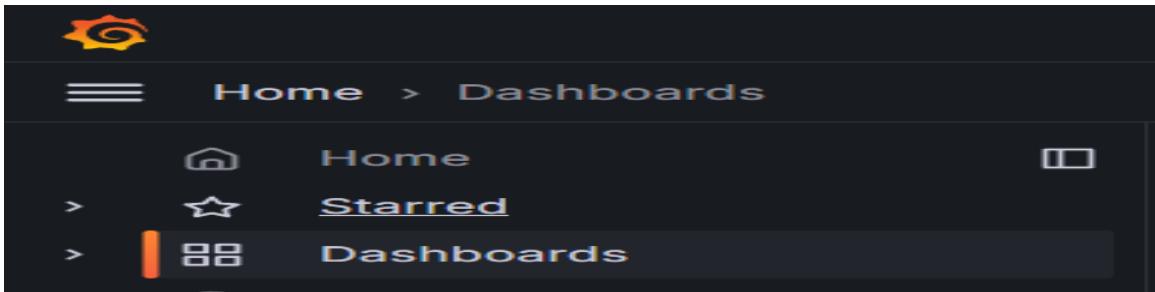
The screenshot shows the 'Data Sources' configuration screen in Grafana. At the top, there is a search bar with the placeholder 'Search or jump to...' and a keyboard shortcut 'ctrl+k'. Below the search bar, the breadcrumb navigation shows 'Sources > prometheus-sumit'. The main configuration card for 'prometheus-sumit' has a 'Name' field set to 'prometheus-sumit', a 'Default' checkbox which is checked, and a 'Status' indicator showing a green checkmark. A note below the card states: 'Before you can use the Prometheus data source, you must configure it below or in the config file. For detailed instructions, [view documentation](#)'. A note also indicates: 'Fields marked with * are required'. The 'Connection' section contains a 'Prometheus server URL *' input field with the value 'http://3.109.2.121:9090/'. The entire configuration card has a dark background.

- **Scroll down and click on save and test**

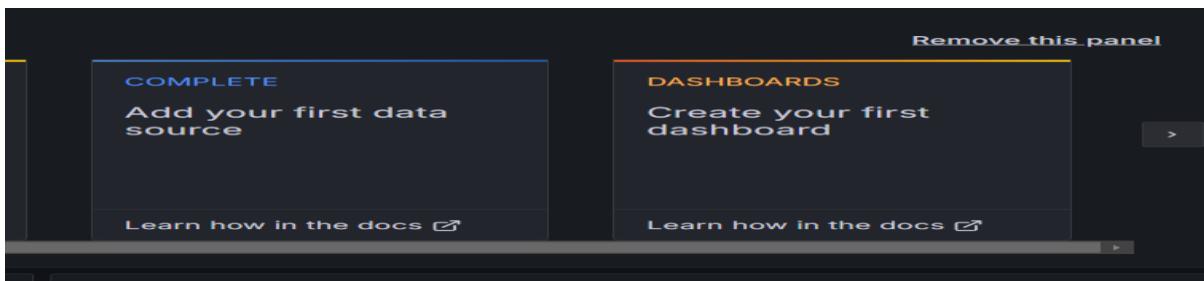
The screenshot shows a confirmation dialog box. It includes sections for 'Other' (Custom query parameters: 'Example: max_source_resolution=5m&timeout='; HTTP method: 'POST'), 'Exemplars' (with a '+ Add' button), and a message area. The message area contains a green checkmark icon followed by the text 'Successfully queried the Prometheus API.' and a note: 'Next, you can start to visualize data by [building a dashboard](#), or by querying data in the [Explore view](#)'. At the bottom, there are two buttons: 'Delete' (red) and 'Save & test' (blue).

- **You can see successfully queried the Prometheus API**

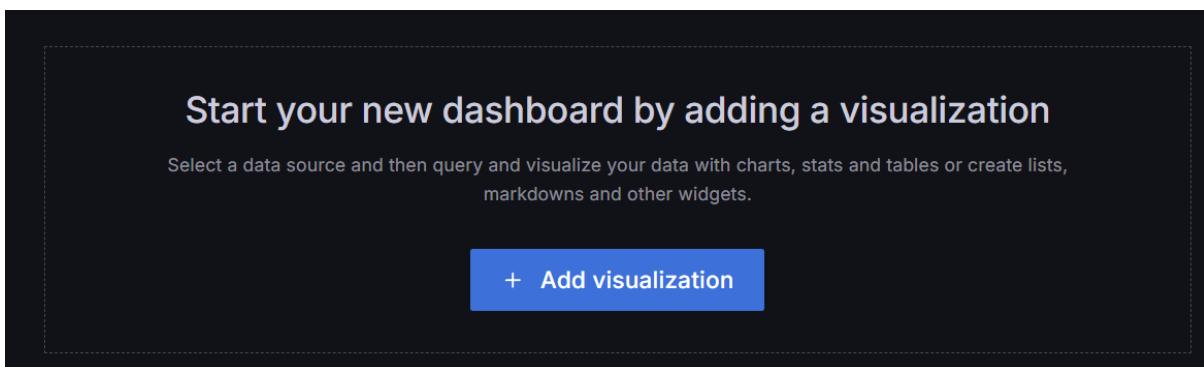
- Now click on Dashboard



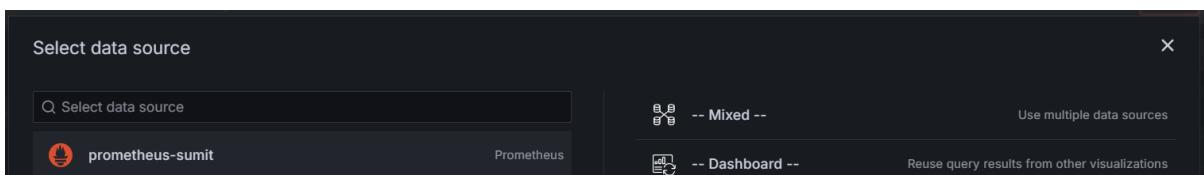
- Now click on create your first dashboard



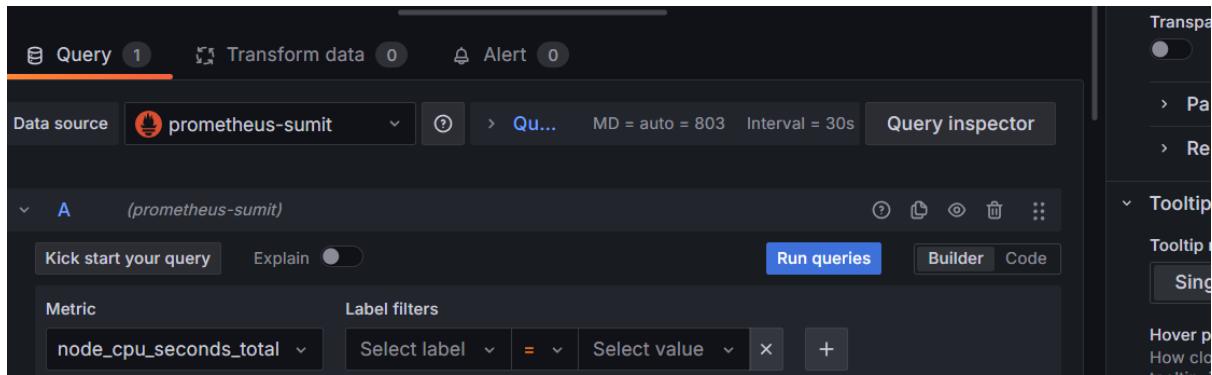
- Then click on add visualization



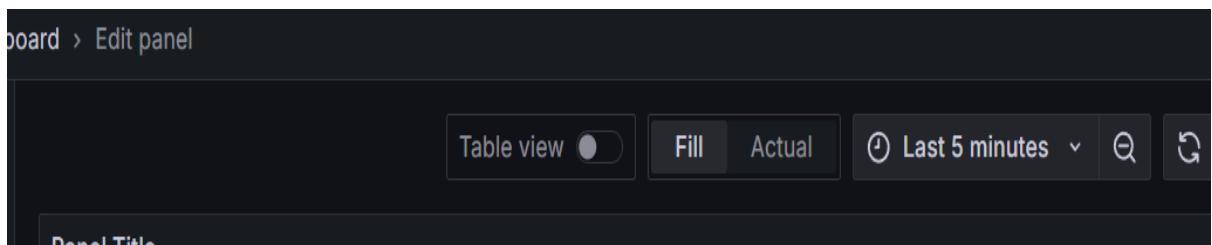
- Select Prometheus-sumit that we have created



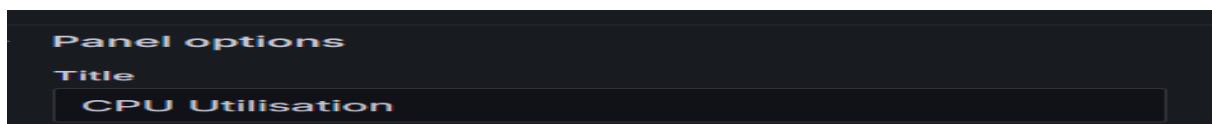
- Select node_cpu_secods_total then click on Run queries



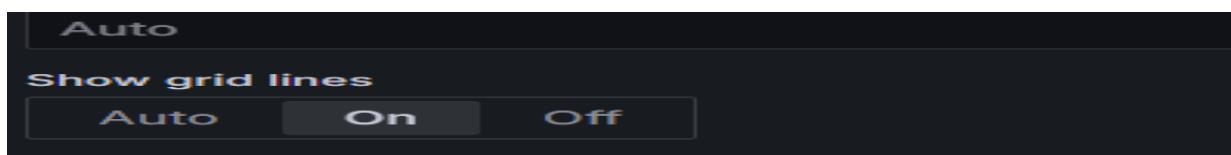
- Select Last 5 minutes



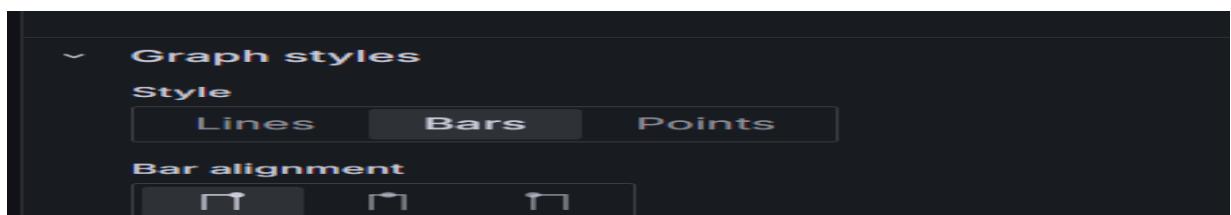
- Given title name as CPU Utilization

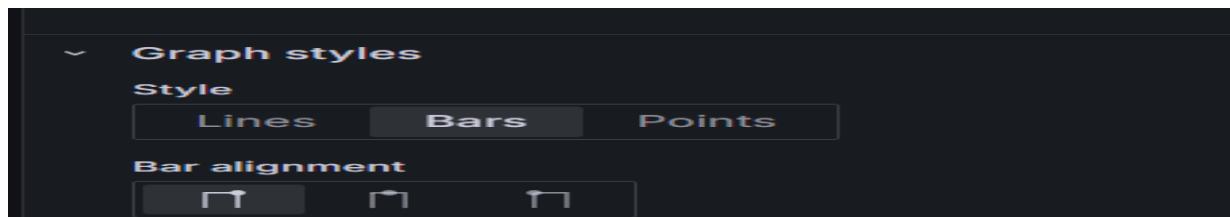


- On the Grid lines

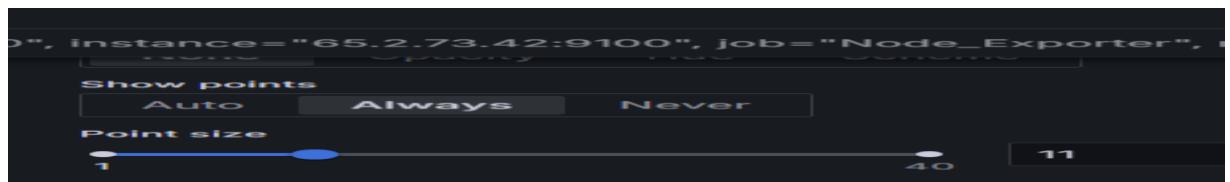


- In style select Bar





- Show point always and Increase point size



- Click on save and Give title then click on save

Save dashboard

New dashboard

Details

Title

Dashboard for Banking-Finance application build server

Description

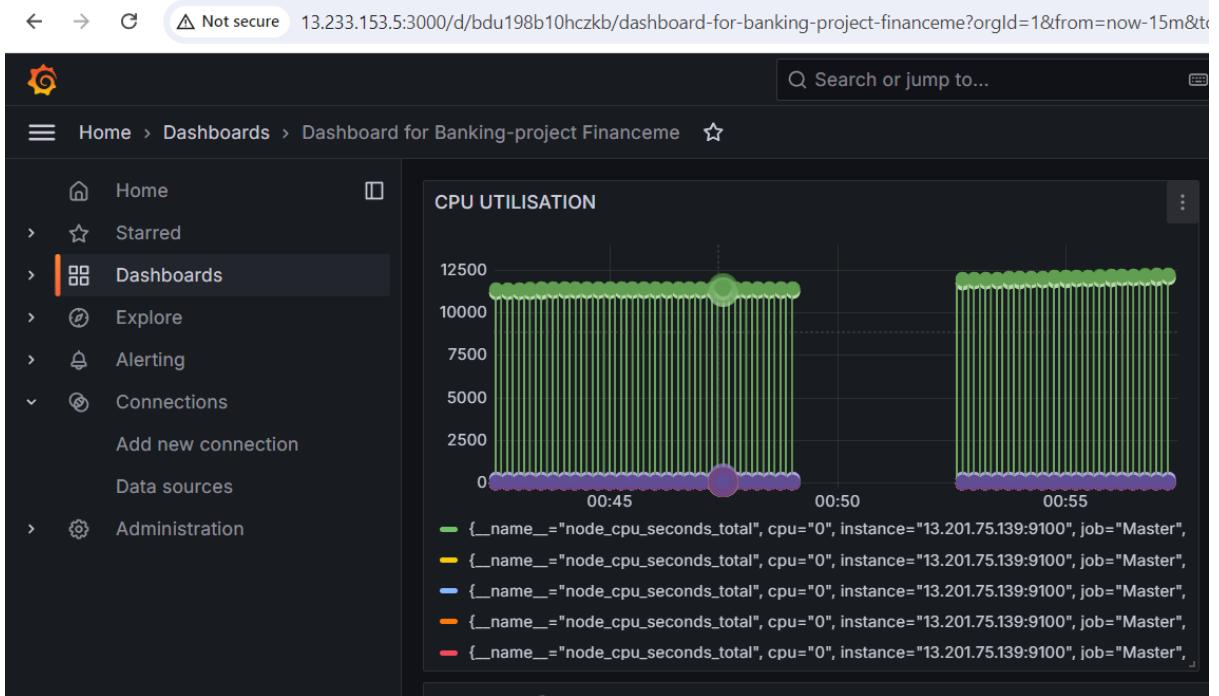
Created by sumit

Folder

Dashboards

Cancel Save

- Now you can see the CPU utilization dashboard for Banking-Finance application build server

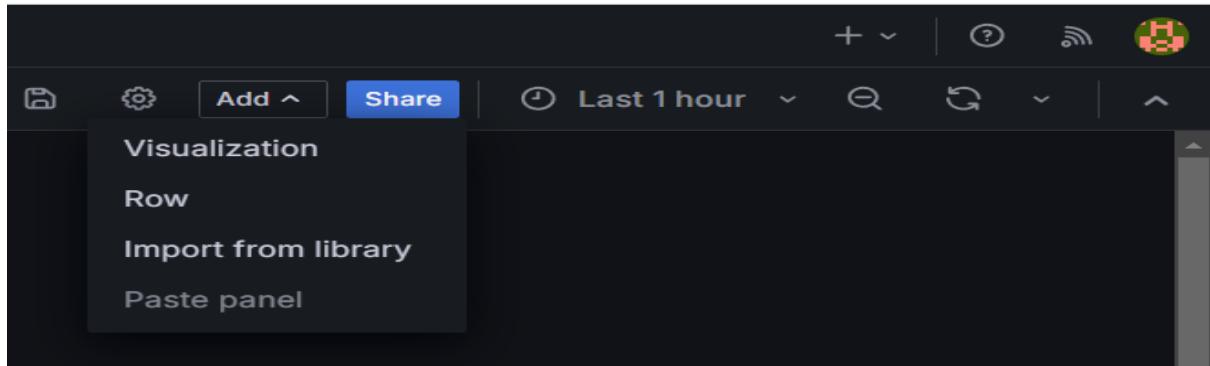


- Now click on add to add another visualization for network
- click on save after create dashboard
- Now you can see the network

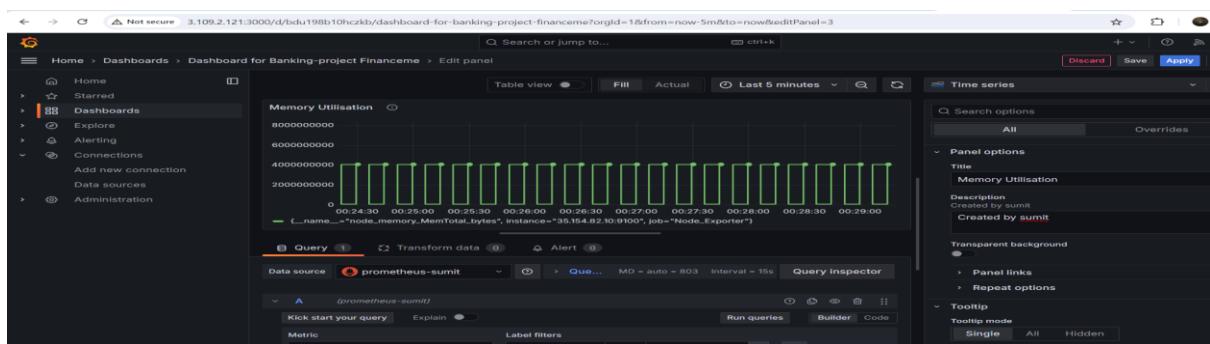


Step 46: Now adding memory utilization in dashboard

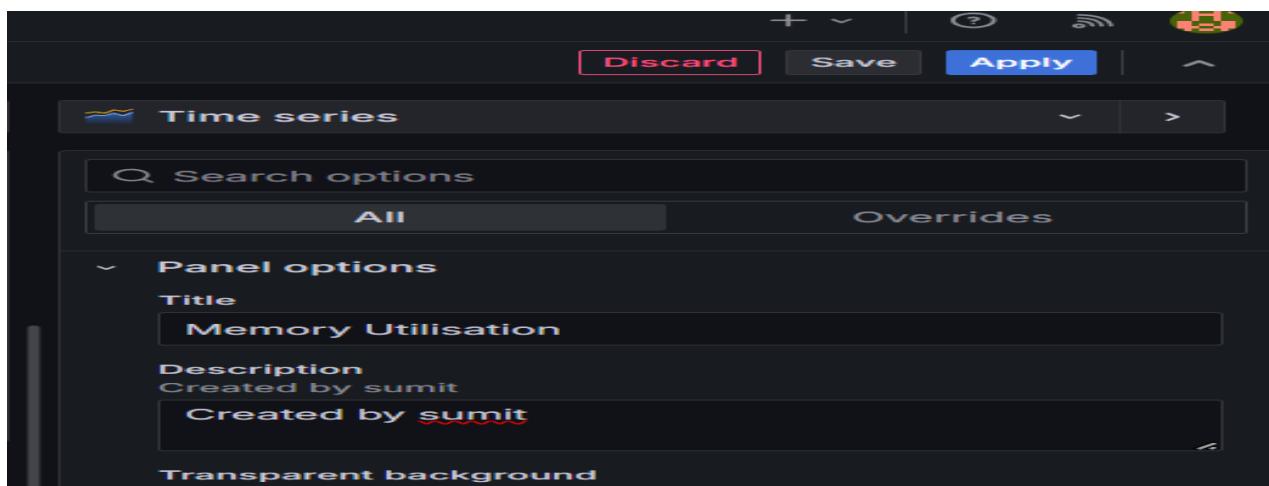
- Click on add



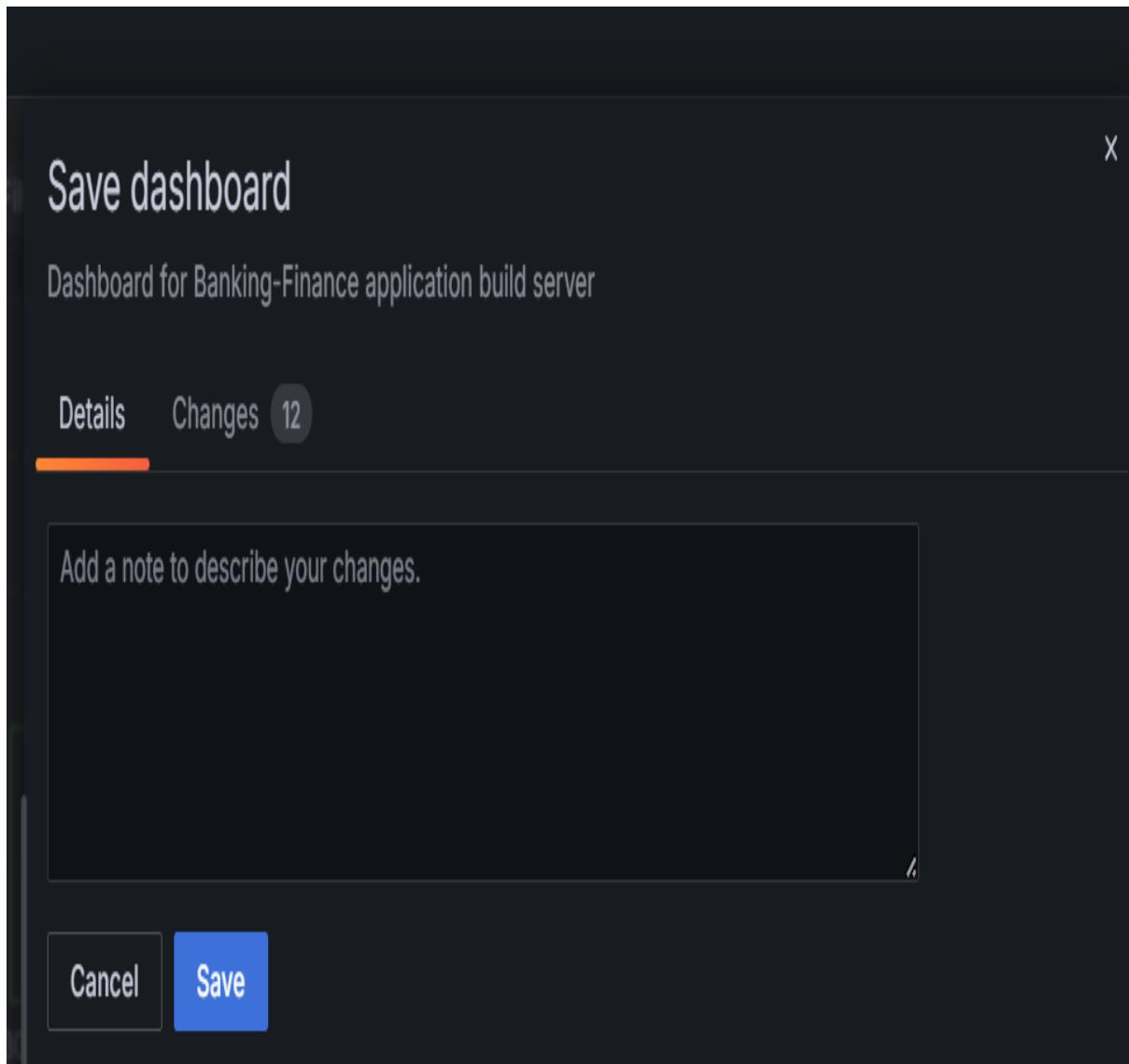
- Then select visualization
- In metrics type: `node_memory_MemTotal_bytes` and click on run queries ,select last 5min



- Give title as Memory utilisation and set every thing as before

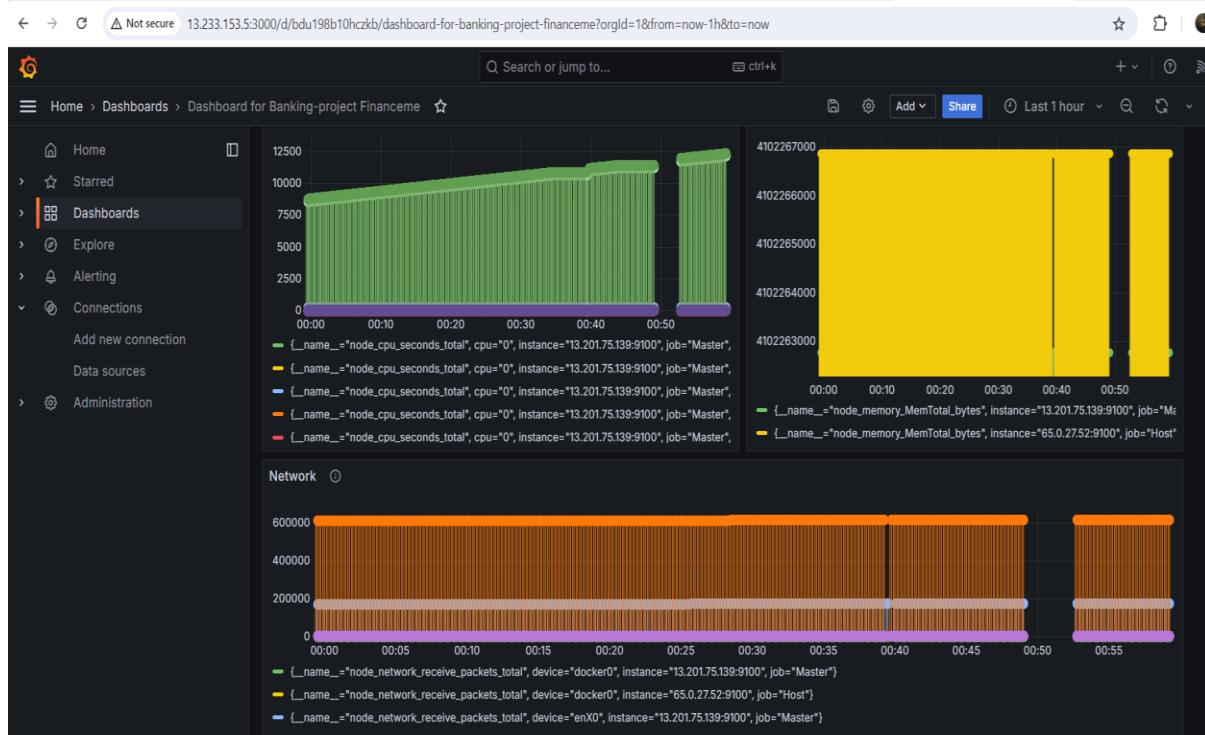


- Click on save

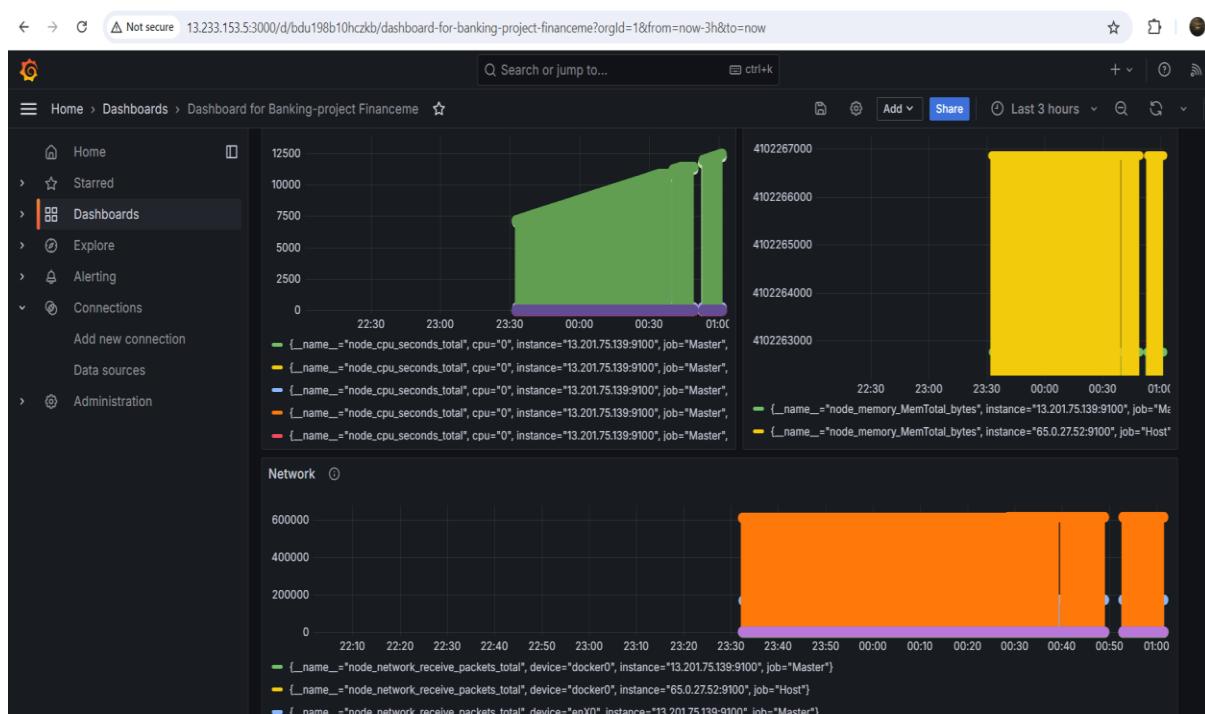


Step 47: Click on dashboard ,You can see we have created a dashboard for monitor CPU utilization ,Memory Utilization, and Network

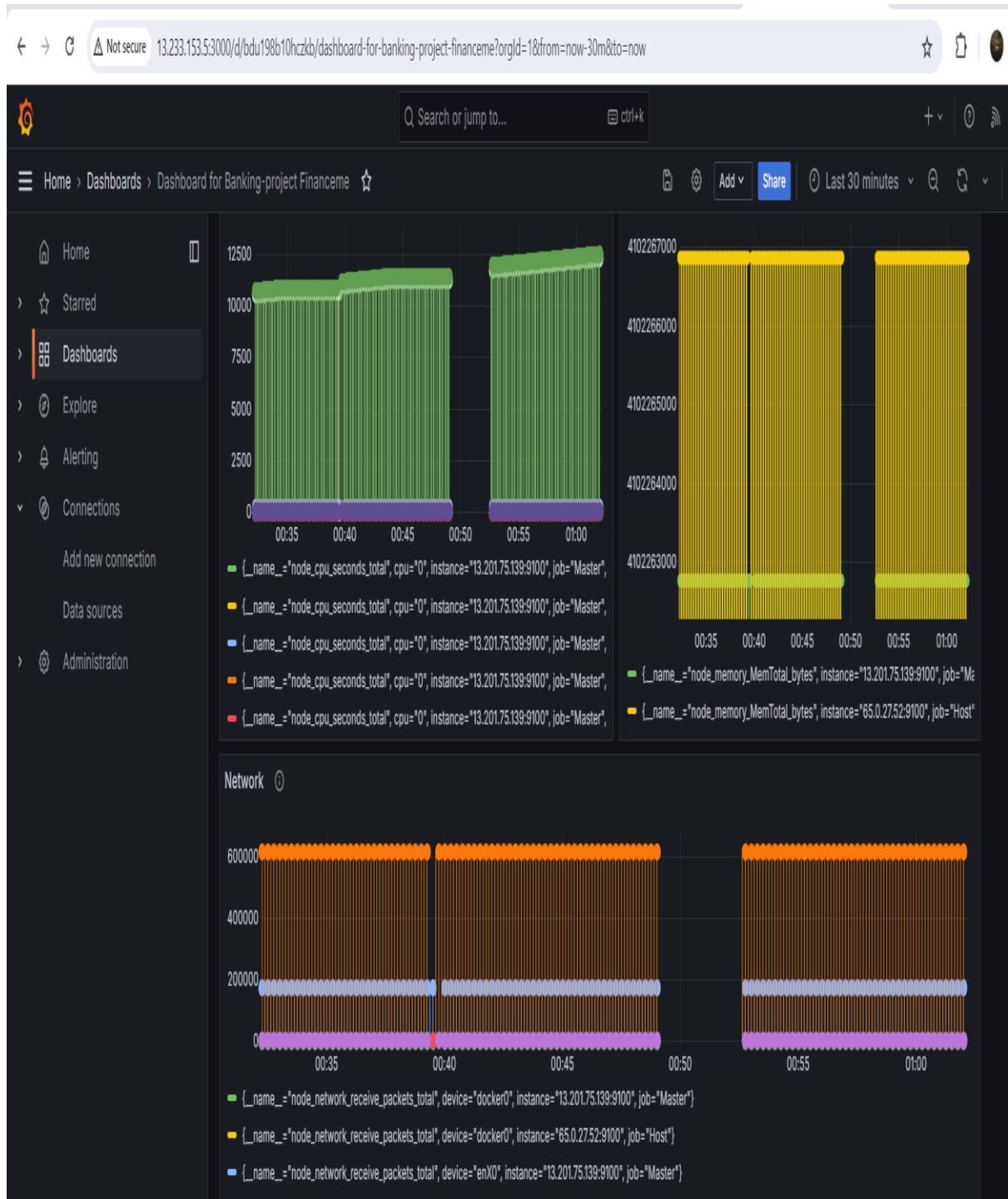
- For last 1 hour-



- For last 3 hours-

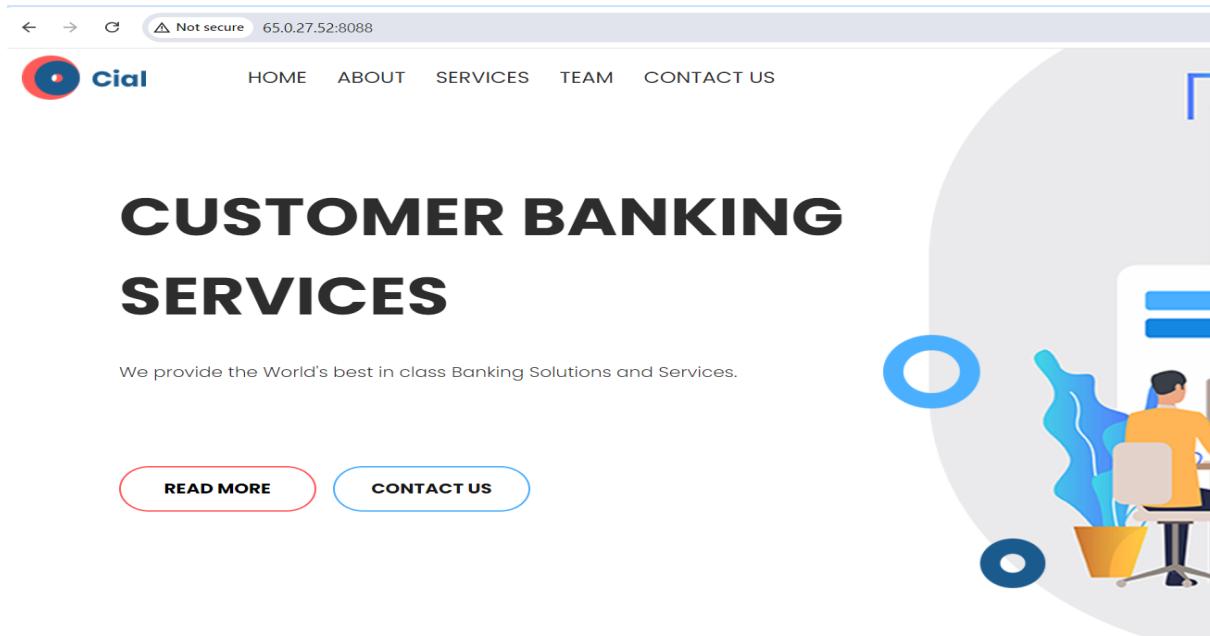


- **For last 30 minutes-**

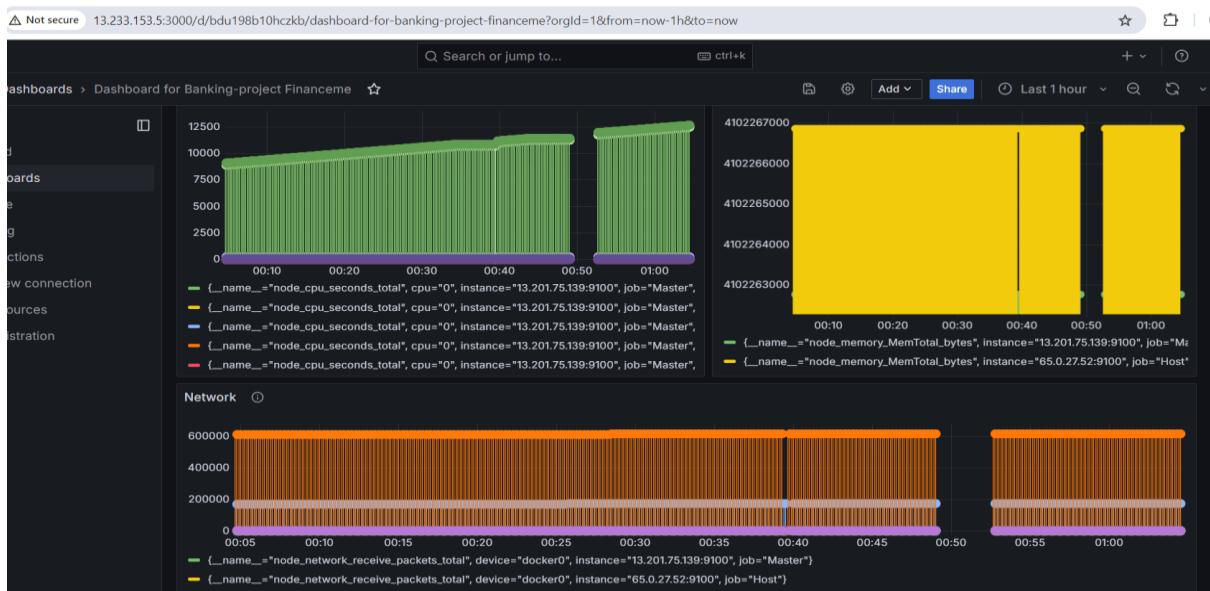


❖ Hence we have created a **Banking_project** web application By using jenkins ,ansible via Jenkins,docker,git,dockerhub etc. and we have also monitored the CPU Utilisation, Memory utilization as well as network of build server

- Banking-project Financeme is accessible-



- Dashboard for monitoring(CPU Utilisation,Network and Memory utilization)



- ❖ Hence we have created a Banking-project web application(financeme),which is accessible on internet we have also created dashboard for CPU Utilization,Network and Memory utilization of build server